TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

BAKALÁŘSKÁ PRÁCE

2012 Martin Sobotka

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B 2612 – Elektrotechnika a informatika

Obor: 1802R007 – Informační technologie

Explorace vnitřního prostředí pomocí platformy IRobot Create

Exploration of the indoor environment using iRobot Create platform

Bakalářská práce

Autor: **Martin Sobotka**

Vedoucí práce: Ing. Jan Strnad

Konzultant: doc. Mgr. Ing. Václav Záda CSc.

V Liberci 17. května 2012

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/ 2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Poděkování

Na tomto místě bych chtěl poděkovat panu Ing. Janu Strnadovi za odborné vedení mé práce. Dále bych chtěl poděkovat panu Doc. Mgr. Ing. Václavu Zádovi CSc. za umožnění přístupu do laboratoře a zapůjčení potřebného vybavení. V neposlední řadě bych rád poděkoval mým přátelům, rodině a kolegům, kteří mě při tvorbě této práce podporovali a motivovali.

Martin Sobotka

Abstrakt

Hlavním cílem této práce bylo vybrat, podrobně otestovat a osadit vhodný dálkový senzor pro platformu iRobot Create. Následně pak vytvořit aplikaci demonstrující využití osazeného senzoru v praxi - tj. vytvořit aplikaci, která bude řídit robota tak, aby se vyhnul překážkám, a nabídne uživateli jednoduchou mapu prostředí, kam překážky zakreslí.

Práce v úvodu představí platformu iRobot Create a navrhne optimální cenově dostupný dálkoměr pro rozšíření jejich možností. Hlavní část práce je věnována rozboru senzoru Microsoft Kinect a zpracování jeho výstupních dat - obrazu. U senzoru je popsán jeho funkční princip a následně je provedena série testů, které se zaměřují na jeho klady a zápory při použití v mobilní robotice. Z výsledků těchto měření vychází závěry, ze kterých je možné určit optimální umístění senzoru i možné operační nasazení. Pro potřeby projektu bylo vytvořeno několik programů, které se zabývají prací se senzorem a zpracováním jeho obrazu (např. nalezení objektů). Poslední aplikace kombinuje využití knihovny ovládající robota spolu s vytvořenou knihovnou pro senzor Kinect a demonstruje možné využití senzoru v mobilní robotice.

Klíčová slova

Senzor Kinect, iRobot Create, mobilní robotika, zpracování obrazu, robotická explorace

Abstract

The main goal of this project was to choose, test and set up a suitable remote sensor for iRobot Create platform. Consequently to create an application demonstrating the usage of a set up sensor in practice - that being, to create an application which would control the robot so it would dodge obstacles and which would offer a simple map of environment where these obstacles are to be drawn for the user.

In the introduction of the writing the iRobot Create platform is presented and an optimally cost-accessible distance meter is suggested for expanding it's capabilities. The main part of the writing is devoted to examination of a Microsoft Kinect sensor and processing of it's output data - the video. The functional principle is described in the sensor part, followed by a series of tests which focus on it's positives and negatives with usage in mobile robotics. From the results of these tests are made conclusions from which it is possible to tell the optimal placement of the sensor and it's possible operation deployment. For the needs of this project, there were created multiple program which are engaged with working with the sensor and processing of it's video (e.g. finding of objects). The last application combines utilising of the library controlling the robot with created library for Microsoft Kinect sensor, and demonstrates possible utilisation of the sensor in mobile robotics.

Keywords

Kinect sensor, iRobot Create, mobile robotics, image processing, robotic exploration

Obsah

Seznam zkratek

IR - Infrared

SDK - Software Development Kit

USB - Universal Serial Bus

RGB - Red-Green-Blue

I2C - Inter-Integrated Circuit

CMOS - Complementary Metal-Oxide-Semiconductor

FAQ - Frequently Asked Questions

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

1 Úvod

Cílem této práce bylo vybrat vhodný senzor pro robotickou exploraci s využitím na platformě iRobot Create. Jádrem projektu je pak test a osazení zakoupeného senzoru na platformu spolu s demonstrační úlohou. Práce se podrobně zabývá senzorem Microsoft Kinect a jeho možnostmi v mobilní robotice. Tato bakalářská práce navazuje na bakalářský projekt, při kterém byla nahrazena většina originálního ovládání platformy - Open Interface. To využívá k řízení robotů bytové sekvence, které nahradila programátorsky přívětivější knihovna s metodami, jež nevyžaduje kompletní znalost původního bytového protokolu. Ovládací knihovna je psaná v jazyce C#, stejně tak tato práce.

Původní motivací pro tento projekt bylo vytvoření knihovny v jazyce C#, která bude schopna ovládat robota a využít ho následně v jiných projektech pro úlohy týkající se plánování (tj. podobor umělé inteligence). Základní ovládání robota bylo zvládnuto již během bakalářského projektu - pro tuto část posloužil jako předloha švýcarský projekt emss (Environment Mapping Self-Sustainable Robot). Jedná se knihovnu v jazyce C++, jež zajišťuje navigaci a vzdálené ovládání platformy iRobot Create.

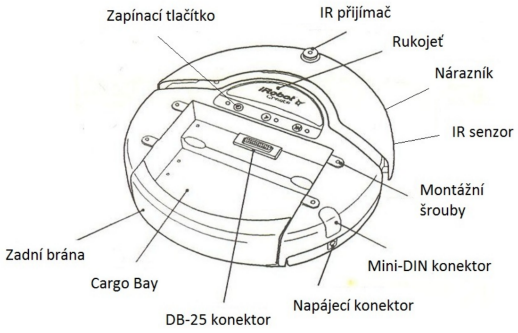
Cílem této práce by také mělo být rozšíření možností robota ve smyslu jeho vnímání okolí pro využití v dalších úlohách, v nichž bude požadována jeho větší autonomie a získání dat o prostoru, ve kterém se robot nachází.

2 Platforma iRobot Create

Firma iRobot, zabývající se různými odvětvími robotiky od mobilní až po vojenskou, je známá zejména výrobou čistících robotů. Nejznámější řadou jsou robotické vysavače iRobot Roomba. Od něj je odvozena platforma Create, která je určena zejména pro výzkumné účely. Je tedy zbavena čistícího modulu a místo něho obsahuje nákladový prostor spolu s dvěma konektory pro komunikaci s dalšími zařízeními a vlastní ovládání.

(a) iRobot Roomba

(b) Platforma iRobot Create



(c) Popis platformy

Obrázek 1: Ukázka robotů firmy iRobot a popis platformy iRobot Create
(obrázek iRobot Roomba převzat z www.productwiki.com).

Kromě bočního napájecího konektoru je tedy přítomen sedmipinový mini-DIN konektor určený pro komunikaci s robotem po sériové lince (s platformou je dodávána redukce z tohoto konektoru na RS232) a 25 pinový konektor pro volné využití s libovolnou periferií. V tomto konektoru se nalézají digitální vstupy a výstupy, analogový vstup, indikační piny (např. robot se nabíjí) atd. Kompletní schéma zapojení konektorů naleznete v dokumentu „iRobot Create Open Interface (OI) Specification“ od firmy iRobot. V něm také naleznete kompletní seznam příkazů, které robot zvládá.

Robot je také vybaven několika senzory převážně zajišťujících jeho pohybovou bezpečnost. K tomuto účelu jsou ve výbavě detektory propadu jednotlivých kol, infračervené - „cliff“ - senzory v krytu nárazníku, které detekují vzdálenost od povrchu (slouží tak jako prevence proti pádu např. ze schodů) a samotný nárazník vybavený tlakovými senzory na bocích. Lze tak určit, z jaké strany robot narazil. Tyto bezpečnostní prvky nechrání robota, pokud couvá. Z tohoto důvodu se tento pohyb obecně nedoporučuje. Dále je robot vybaven IR přijímačem pro detekci dokovací stanice a bočním IR senzorem pro detekci stěny. O něm podrobněji v následující kapitole.

3 Senzory

Platforma iRobot Create nedisponuje v základu vhodnými senzory pro robotickou exploraci vnitřních prostor. Jediný integrovaný senzor, který lze pro danou problematiku využít je IR senzor na pravém nárazníku s dosahem přibližně 15 cm.

Obrázek 2: IR senzor pro sledování stěny na nárazníku robota.

Tento senzor je primárně určen pro algoritmy sledování stěny, tj. sledování stěny po boku robota bez zbytečných nárazů a s minimální odchylkou. Robot obsahuje tento algoritmus jako jedno z integrovaných demo programů. Demo je funkční, nicméně nedokonalé. Pokud robot mine ostrou pravou hranu (příkladem ohyb stěny o 90°), často se stěnou ztratí kontakt a přejde do jízdy v mírném oblouku, dokud nenarazí na další stěnu. Pokud bychom chtěli využít tento specifický pohyb (tj. sledování stěny) pro mapování prostoru, jsou tyto přejezdy značnou komplikací. Jestliže má robot mapovat překážky, oblast za pravou zatáčkou úplně mine. Proto byl již během bakalářského projektu vytvořen a nasazen do praxe alternativní algoritmus, který tento problém přejezdů rohů úspěšně řeší.

Za předpokladu, že budeme zvažovat snahu o co nejrychlejší a nejpřesnější mapu prostředí, tento senzor samotný k vyřešení problematiky nestačí. Uvedeným senzorem lze získat hrany místnosti, případně jiného objektu, který lze bočně objet. Objekty mimo (příkladem stůl umístěný uprostřed místnosti) zůstanou nezaznamenány.

Z uvedených důvodů byla jedním z prvních úkolů rešerše se zaměřením na výběr nejvhodnějšího doplňkového senzoru s nízkou cenou pro zjednodušení a zefektivnění průzkumu okolí. Výsledky jsou stručně popsány v podkapitole 3.2. Jako nejvhodnější senzor pro záměry této práce byl vybrán ultrazvukový dálkoměr SFR08. O tento senzor bylo také zažádáno, ale po diskuzi Technická univerzita v Liberci zakoupila výrazně dražší a efektivnější pohybový senzor Microsoft Kinect, který původně nebyl vzhledem k požadavku na cenu v rešerši zvažován. O pohybovém senzoru Microsoft Kinect se více dočtete v kapitole 4.

3.1 Dálkoměry

Dálkoměry jsou nezbytnou součástí projektů zabývajících se robotickou explorací prostředí. Umožňují robotovi získat údaje o vzdálenosti objektů v jeho okolí, což je nezbytné pro úlohy určitého autonomního chování (např. objíždění překážek).

Pokud budeme uvažovat o senzorech v nižších cenových kategoriích (tj. maximálně 1500Kč), připadají v úvahu senzory na principu IR (podkapitola 3.1.2) a ultrazvukového sonaru (podkapitola 3.1.1). Dražší laserové dálkoměry měří vzdálenost pomocí rozdílu času nebo vlnové délky mezi vyslaným a přijatým paprskem. Vzhledem k ceně nebyly tyto dálkoměry pro práci zvažovány.

3.1.1 Ultrazvukové senzory (sonar)

Ultrazvukové senzory pracují na principu odrazu akustického signálu. Nevrátí přesnou polohu předmětu, ale pouze vzdálenost, ve které se předmět nachází. Jejich funkčnost a přesnost mohou ovlivnit např. teplota a hustota okolního prostředí (vzduchu), materiály, jež zvukové vlny pohlcují, nebo i rychlost vysílače vůči přijímači (Dopplerův efekt). I tak ale nabízí vysokou spolehlivost u většiny objektů (překážek) a patří mezi hojně využívané senzory, a to nejen v mobilní robotice.

V této kategorii se rozhodovalo mezi třemi senzory: SFR02, SFR08 a SFR10. Senzory se připojují přes sériovou linku (SFR02) nebo sběrnici I2C. Robot v základu nedisponuje možností ovládat jiná zařízení přes tyto sběrnice. Pro obě řešení ale existuje cenově dostupný převodník na USB, přes který by bylo možné senzory propojit k ovládacímu počítači na robotovi.

Tabulku s nejdůležitějšími vlastnostmi uvažovaných senzorů naleznete v příloze A. Z uvažovaných sonických senzorů byl jako nejvhodnější pro řešení zadané problematiky zvolen senzor SFR08. Má výborné spektrum dosahu s možností detekce více objektů v různých vzdálenostech, menší vyzařovací úhel než SFR10 a jako nadstavbu obsahuje světelný detektor, který by bylo možné využít k automatickému ovládání přisvětlujících diod. Komunikace senzoru probíhá po sběrnici I2C.

3.1.2 Infračervené „IR“ senzory

Infračervené senzory měří přesnou vzdálenost jednoho bodu. Jednodušší (odrazové) verze těchto senzorů pouze oznámí, zda se před ním nachazí nějaký objekt. Složitější (dálkové) pak vrátí určitou hodnotu, ze které lze vyčíst reálnou vzdálenost. Podstatnou nevýhodou těchto typů může být závislost na povrchu zkoumaného objektu - senzory může ovlivnit tvar, barva či materiál.

Výhodou pro toto zadání je, že lze infračervený senzor připojit přes analogový vstup v nákladním prostoru robota.

Tabulku s nejdůležitějšími vlastnosti uvažovaných senzorů - GP2Y0A02 a GP2Y0A21 - naleznete v příloze A. Infračervený senzor není vhodný jako primární dálkoměr pro zadanou úlohu vzhledem k možné chybě při měření kvůli špatnému odrazu a také kvůli samotnému principu měření. Pokud by byl robot vybaven jedním tímto senzorem, bylo by složité a časově náročné jedním měrným bodem prozkoumat celou zadanou scénu.

3.2 Výběr dálkoměru pro danou problematiku

Jako nejvhodnější primární senzor byl z uvedených zvolen senzor ultrazvukového typu. K tomuto rozhodnutí vedlo zjištění, že infračervené senzory jsou sice schopny přesně změřit vzdálenost daného bodu, nicméně jejich nevýhodou je náchylnost na tvar a materiál odrazové plochy (tj. předmětu). Sonický senzor oproti tomu nezjistí přesnou polohu předmětu, za to spolehlivě změří vzdálenost, ve které se předmět nachází. Po prostudování technických specifikací jednotlivých senzorů byl vybrán jako nejvhodnější ultrazvukový senzor SFR08 s možností doplnění o infračervený senzor GP2Y0A21. Záměrem bylo využít sonický senzor pro detekci prostředí před robotem (tím vytvořit hrubou mapu) a infračervený (s delším dosahem než poskytuje integrovaný v nárazníku) na následnou přesnou detekci vzdálenosti objektů od boku robota.

Po diskuzi s Technickou univerzitou v Liberci byl nakonec zakoupen technologicky mnohem novější senzor, jehož možnosti pro mobilní robotiku nejsou v současné době oproti ultrazvukovým a IR senzorům zdaleka tak podrobně probádané. Tento senzor nebyl v rešerši vzhledem ke své vyšší ceně zvažován - jedná se o Microsoft Kinect a jeho podrobnému rozboru se věnuje následující kapitola.

4 Microsoft Kinect

Po zakoupení Kinectu z fondů Evropské unie v prosinci 2012 se jedním z nejdůležitějších cílů práce stala analýza schopností a nedostatků tohoto senzoru při využití v mobilní robotice. Pro senzor v mobilní robotice jsou důležitými parametry mimo jiné přesnost a schopnost korektně interpretovat různé objekty (překážky). To vedlo k sérii měření a testů. Kompletní soubor fotografií a měření se nachází na přiloženém cd, přímo v práci jsou uvedena pouze nejpodstatnější data pro podporu závěrů.

V kapitole je stručně zachycena historie zařízení, jednotlivé komponenty, jaké ovladače jsou k dispozici, odůvodněný výběr ovladačů a jednotlivé testy zařízení, skladající se z testů měřitelné oblasti, přesnosti měření, prostředí, ve kterém může být senzor používán, minimálních zaznamenaných objektů a pozice umístění senzoru na cílového robota.

4.1 Historie zařízení

Senzor Kinect, při vývoji označovaný Microsoftem jako „projekt Natal“, je založený na technologii vyvinuté firmou PrimeSense v roce 2005 (více v následující podkapitole). Projekt Natal poprvé Microsoft představil na výstavě „Electronic Entertainment Expo“ konané 1. června 2009. Na trh byl jako herní ovladač pro konzoli Microsoft Xbox 360 (již pod názvem Microsoft Kinect) uveden 4. listopadu 2010[1]. Verze pro Windows byla uvedena 1. února 2012.

4.2 Technologie

Obrázek 3: Původní zařízení vyvinuté firmou PrimeSense (převzato z www.hizook.com).

Senzor Kinect je založen na výše zmíněném návrhu firmy PrimeSense. Obrázek 3 zobrazuje systém založený na jednom čipu, který ovládá infračervený emitor, dvě kamery (jednu obrazovou RGB, druhou snímající infračervené záření, obě s maximálním rozlišením 640x480px typu CMOS) a pole

[1]Senzor se zapsal do Guinessovy knihy rekordů jako nejrychleji prodávaný produkt v oblasti spotřební elektroniky, když se ho za 60 dnů prodalo přes 8 milionu kusů.

mikrofonů pro zjištění polohy mluvčích.

K získání hloubky je využita kombinace infračervené kamery a emitoru - emitor nasvítí prostor infračerveným světlem (vlnová délka 950nm) a kamera zaznamená nasvícené objekty. Zpracováním těchto dat lze získat poměrně přesný obraz hloubky prostoru.

Existuje několik omezení daných technologií. Nevýhody infračervených senzorů jsou stručně popsány v kapitole 3.1.2. Zde platí podobné podmínky - plocha musí být nasvítitelná a odrazivá. Předměty jako skleněné desky (obrázek 10) jsou tímto senzorem velmi zřídka interpretovány korektně, stejně tak jako tenčí kulaté objekty (např. železná tyč s průměrem 6 mm, viz kapitola 4.5.1). Tyto oblasti jsou převážně označeny jako „neplatná data“. Senzor je též vybaven tříosým akcelerometrem, barevnou diodou pro zobrazení statusu zařízení a motorem, který ovládá náklon zařízení (rozmezí ±27°). Senzor se k nové verzi konzole Xbox 360 připojuje pomocí speciálního konektoru, stará verze konzole a osobní počítače musí využít přikládanou redukci na USB s napěťovým adaptérem.

4.3 Parametry jednotlivých komponent

Obrázek 4: Nejdůležitější hardwarové části senzoru Kinect (převzato z www.3dnews.ru).

Nápověda Microsoftu pro Kinect obsahuje následující data (přeloženo z anglické dokumentace):

Tabulka 1: Hardwarové parametry Kinectu

Pozorovací úhly	43° vertikálně, 57° horizontálně
Motorizovaný vertikální náklon senzoru	±27°
FPS obou kamer	30 (při rozlišení 640x480px)
Audio formát	16kHz, 16bit mono PCM (pulzně kódová modulace)
Audio vstup	pole čtyř mikrofonů s 24bit analog-digitálním převodníkem s možností redukce šumu a ozvěny

4.4 Ovladače

V době tvorby této práce byly pro Kinect k dispozici tři různé ovladače, a to: CL NUI Platform od Code Laboratories, OpenKinect od open source komunity a oficiální ovladače Microsoftu - Microsoft Kinect SDK. Další programy a knihovny (např. AForge.NET) využívají uvedené ovladače, z velké většiny freenect od OpenKinect.

Ovladače a ukázkové aplikace byly testovány na osobních počítačích ASUS Eee PC 1001px („netbook“ - současný ovládací počítač robotů) s Intel Atom N450 a ASUS M50vc as001c s Intel Core2Duo P8400.

Následují klady a zápory jednotlivých ovladačů, snímky aplikací naleznete v příloze D.

4.4.1 CL NUI Platform

Klady:

- Umožňuje přístup ke všem komponentám Kinectu s výjimkou audia (tj. stream kamer, data z akcelerometru, ovládání motoru a diody).
- Nejnižší využití CPU netbooku z testovaných ovladačů.

Zápory:

- Nepodporuje audio.
- Zapouzdření využívá uzavřené knihovny („blackbox“), ke kterým není dokumentace.
- Poslední aktualizace projektu proběhla 10. 12. 2010 - i přes informace na stránkách, že audio „bude brzy implementováno“, je s největší pravděpodobností vývoj těchto ovladačů ukončen.
- Nelze v současné době ověřit funkčnost s novým Kinectem pro Windows.

4.4.2 OpenKinect

Klady:

- Umožňuje přístup ke všem komponentám Kinectu s výjimkou audia (tj. stream kamer, ovládání motoru a diody, data z akcelerometru).
- Projekt vytvářený jako open source (licence Apache 2.0 a GPL v2).
- Projekt nabízí kvalitní veřejnou dokumentaci.

Zápory:

- Nepodporuje audio.
- Nelze zaručit funkčnost v závislosti na hardware (chyba se jeví jako bílá oblast místo videa).
- Nelze v současné době ověřit funkčnost s novým Kinectem pro Windows.

Tento projekt je psaný v jazyce C, na rozdíl od této bakalářské práce, která je vytvářena v C#. Bylo tedy potřeba využít C# wrapperu, který OpenKinect nabízí (k dispozici i pro další jazyky, např. matlab, python). Bohužel již na webových stránkách projektu je naznačeno, že s ním mohou vzniknout pod operačním systémem Windows problémy. Dalším zmiňovaným problémem je kompilátor vývojového prostředí Visual Studio, který v některých případech může mít problém s interpretací kódu (konkrétně jsou zmiňovány explicitní pointery).

Tyto ovladače jsou volně dostupné, pro jejich využití je ale nutné provést kompilaci zdrojového kódu a vytvořit knihovnu freenect, která posléze v programu zajišťuje komunikaci se senzorem. Kompilace probíhala dle návodu na webu projektu. OpenKinect ve verzi 3629b75 se sice podařilo zkompilovat, nicméně ze základních ukázkových aplikací v jazyce C fungovala pouze jedna, a to získání hloubkového obrazu. Další zobrazovaly bílé okno. Stejný výsledek se dostavil i u wrapperu C#. V sekci FAQ je upozorněno na fakt, že se v takovém případě nemusí jednat o chybu kompilace, ale o problémy s určitým hardwarem - zejména s grafickými kartami. Že se jedná o tento problém naznačovalo i vytížení procesoru (přes 95%). Po třech různých kompilacích se stejným výsledkem se od dalšího testování C# wrapperu OpenKinect upustilo.

AForge.NET - Kinect namespace

Jak bylo zmíněno v úvodu kapitoly, OpenKinect implementovalo několik zajímavých projektů. Ty většinou se svou distribucí dodávají i zkompilovanou (někdy upravenou) knihovnu freenect, se kterou je aplikace odzkoušena. 28. února 2012 se implementace objevila i u frameworku AForge.NET (ve verzi 2.2.4; framework zabývající se počítačovým viděním a umělou inteligencí), který byl použit u projektu vzdáleného ovládání platformy iRobot Create. Bohužel testovací aplikace s přibalenou

knihovnou nešla na poslední dostupné verzi ovladačů spustit. Po výměně přibalené knihovny freenect za pro projekt zkompilovanou verzi původně určenou pro C# wrapper OpenKinectu, aplikace začala na netbooku korektně fungovat - z toho lze s jistotou usoudit, že chyba byla opravdu zapříčiněna způsobem vykreslování, který OpenKinect ve svých ukázkových aplikacích využívá.

AForge.NET poskytuje v projektu přístup ke všem komponentám Kinectu s výjimkou audia (stejně jako OpenKinect, který využívá). Využití procesoru ukázkovou aplikací AForge se zkompilovanou knihovnou freenect je přibližně 95%. Aplikace byla mírně upravena pro záznam obrazu a následně využita pro získání dat k porovnání s údaji, které poskytuje Microsoft Kinect SDK po postprocesingu.

4.4.3 Microsoft Kinect SDK

Klady:

- Oficiální podpora výrobce (a to i pro nový Kinect pro Windows, vydaný 2.1.2012).
- Jako jediný podporuje audio.
- Projekt nabízí kvalitní veřejnou dokumentaci.
- Automatický postprocesing vyhledá v obraze hráče a vrátí 20 bodů těla (2 aktivní, celkem až 6 evidovaných osob).

Zápory:

- Neumožňuje přístup k akcelerometru a nastavení diody.
- Změna náklonu (motorem) pozastaví streamy kamer.
- Vysoké systémové požadavky.
- Hlášené chyby SDK (např. start sledováním kostry zruší aktivní audio stream).

Po obdržení senzoru bylo započato testování v poslední dostupné verzi SDK s ovladači. Těmi v prosinci 2011 byly KinectSDK-v1.0-beta2. Minimální systémové požadavky jsou dvoujádrový procesor na frekvenci 2.66GHz nebo výkonnější, 2 GB RAM a nesdílený port USB 2.0 - konektor nesmí být sdílený s webkamerou nebo mikrofonem, jinak Kinect nebude fungovat. Netbook tyto specifikace nesplňuje a jeho procesor je neustále vytížený na 100%.

1. února 2012 spolu s novým Kinectem pro Windows (viz kapitola 4.5) vyšla stabilní verze ovladačů, verze 1.0. Oproti testovací verzi beta2 bylo odstraněno několik chyb, pozměněny licenční podmínky, sjednoceny jednotlivé části SDK (audio a video), změnil se způsob zpracovávání příchozích dat, zrychlil se postprocesing, ale některé chyby (např. zrušení audio streamu aktivací vyhledání kostry) zůstaly.

SDK se primárně zaměřuje na vyhledávání osob v obraze, poskytuje ale i další funkce, které mohou být užitečné i při projektech s mobilními roboty. Jako příklad lze jmenovat přemapování hloubkových dat pod obraz z RGB kamery - této funkce lze využít pro intuitivní vybírání objektů přes barevný obraz apod.

4.4.4 Výběr ovladačů

Výběr ovladačů je vždy jedním z klíčových kroků vytváření aplikace. V tomto případě se rozhodovalo mezi:

- starými, pravděpodobně již v budoucnu neaktualizovanými ovladači, které však byly při testování nejméně náročné na hardware;
- pravidelně aktualizovanými ovladači open source, které zpřístupní všechny funkce Kinectu (s výjimkou audia), ale jejichž provozování pod Windows by mohlo být problematické;
- oficiálními ovladači Microsoftu, které mají vysoké hardwarové požadavky, neumožní přístup k akcelerometru, ale u kterých by měla být zaručena nejlepší podpora a kompatibilita se zařízením.

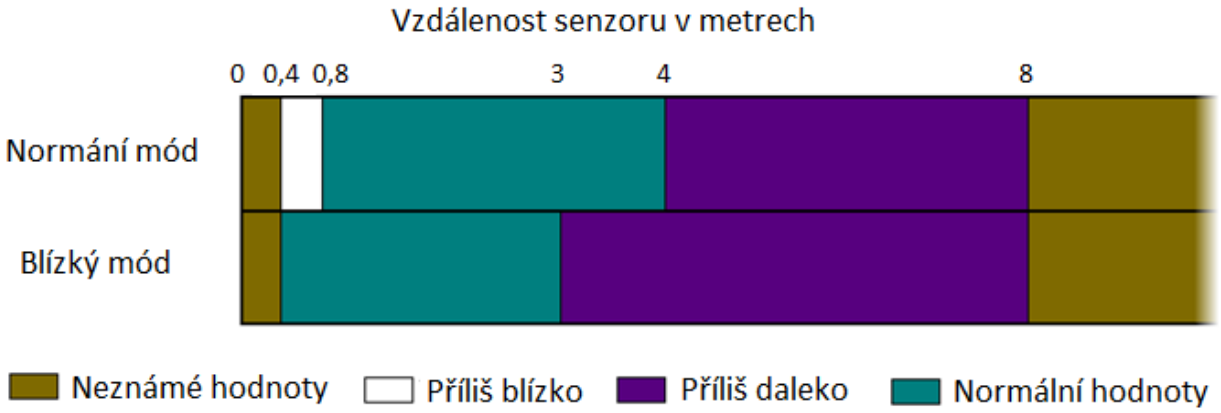
I když AForge v kombinaci s OpenKinectem poskytuje nižší nároky na hardware díky chybějícímu postprocesingu, po vydání SDK 1.0 Microsoftem se rozhodlo o přepsání hlavní aplikace pod tyto ovladače. Hlavním důvodem bylo uvedení nové verze Kinectu - „Kinect pro Windows“, který byl představen spolu s SDK 1.0. Microsoft však posléze oznámil, že toto zařízení má přepracovaný firmware, a z toho lze soudit, že OpenKinect bude mít s podporou problémy (data o testování kompatibility nebyla v dané době dostupná). Novým omezením napovídal i fakt, že se ve výčtovém typu „KinectStatus“ (který se v Microsoft Kinect SDK stará o stav zařízení), nově objevily položky „neověřené zařízení“ a „nepodporované zařízení“. Další testy jsou tedy zaměřeny hlavně na zpracování dat s Microsoft Kinect SDK.

4.5 Měřitelná oblast

Kinect byl do vydání Kinectu pro Windows cílen jako herní ovladač konzole Microsoft Xbox360 a tomu odpovídá i SDK, kde je kladen vysoký důraz na rozpoznání osob v obraze, jejich sledování a predikci jejich pohybů. Samotný obraz hloubky je u původní verze Kinectu prakticky pouze nutný doplněk.

Microsoft Kinect SDK od verze 1.0 spolu s uvedením Kinectu pro Windows možnosti využití pro mobilní robotiku zlepšuje. Nově se zde objevuje tzv. „blízký mód“ dosahu - tj. snímání vzdálenosti

již od 40 cm místo 80 cm. Oproti tomu je maximální rozpoznaná vzdálenost snížena na 3 metry namísto 4 m a rozpoznání kostry nevrací všech 20 bodů těla. Blízký mód je tak pro vnitřní užití robotického průzkumu mnohonásobně vhodnější - robot nepotřebuje rozpoznávat pohyby člověka a maximální dosah 3 m nebo 4 m není tak markantním rozdílem jako minimální dosah 40 cm nebo 80 cm.



Obrázek 5: Dosah Kinectu (převzato z nápovědy Microsoft Kinect SDK)

SDK vrací jako interpretaci hloubkového obrazu pole hodnot datového typu short. Jednotlivé položky pole interpretují vzdálenost v milimetrech a zároveň je v nich uložena informace, zda daný voxel (tj. nejmenší prvek třídimenzionálního obrazu) odpovídá určité osobě. Získání dat prezentuje následující ukázka kódu.

Zdrojový kód 1: Získání informace o hráči a vzdálenosti

```
1  //DepthImageFrame je vnitřní výčtový typ Microsoft Kinect SDK
2  int hráč = příchozi_pole[index] &
       DepthImageFrame.PlayerIndexBitmask;
3  int realna_hloubka = příchozi_pole[index] >>
       DepthImageFrame.PlayerIndexBitmaskWidth;
```

Pokud je proměnná hráč vyšší jak 0, odpovídá daný voxel poloze určitého hráče (osoby). Kinect dokáže v základním modu detekovat až šest osob zároveň, aktivní sledování kostry ale může být aplikováno pouze pro dvě osoby (tzn. postprocesing určující 20 bodů těla).

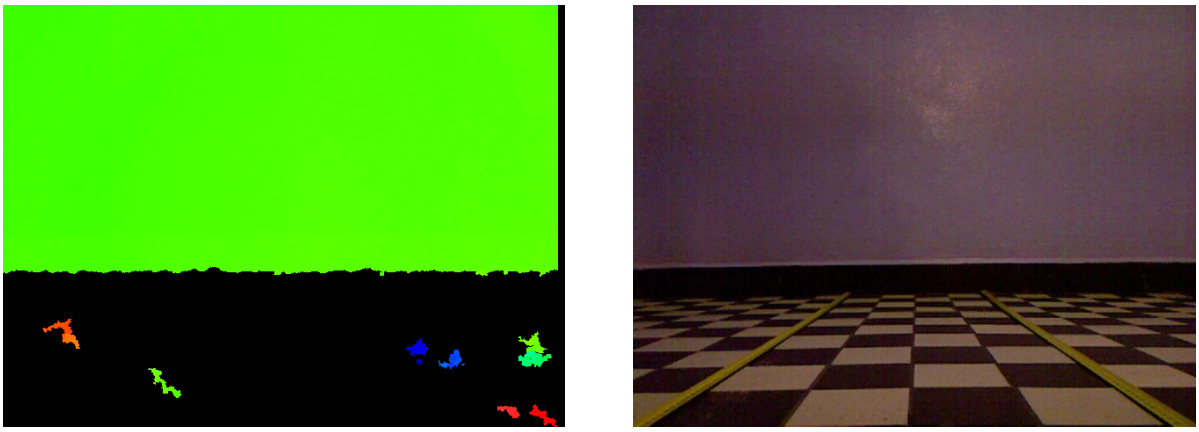
Samotná hodnota reálné hloubky u Kinectu pro Windows závisí na již zmíněném zvoleném modu. Kinect pro Xbox 360 (na kterém je založen tento projekt) je neměnitelně ve výchozím modu a jeho minimální interpretovaná hodnota je udána jako 800, maximální 4095 (mělo by odpovídat vzdálenosti v mm). Reálností návratových hodnot se více zabývá další kapitola s názvem Přesnost.

Zajímavý problém nabízí horizontální šířka obrazu (vertikální v této práci nebylo třeba testovat). Jak se dočtete v kapitole 4.3, oficiální nápověda poskytuje údaje 43° vertikálně, 57° horizontálně pro obě kamery. Objekt „KinectSensor“ (který reprezentuje senzor v SDK) po připojení Kinectu vyčte hodnoty daného zařízení také. Nejen, že jsou hodnoty různé pro každou kameru, ale dle orientačního měření není zcela korektní ani oficiální nápověda, ani vyčtení hodnot v SDK. Jak dokazuje následující obrázek, senzor, který byl při tvorbě práce k dispozici, hlásí namísto 57° pro barevnou RGB kameru 62° a pro infračervenou dokonce 117°. Provedené orientační měření ukázalo, že horizontální pole RGB kamery skutečně odpovídá přibližně 62°, pole infračervené je oproti ní ale menší - v provedeném orientačním měření bylo dosaženo výsledku odpovídajícího 59°.

(a) RGB kamera (b) IR - Kinect SDK (c) IR - OpenKinect

Obrázek 6: Orientační měření zobrazitelného horizontálního úhlu. Na barevném snímku je vidět páska, která svírá úhel 59°.

Zásadní problémem senzoru jsou vady v obraze, tj. nepředvídatelné falešné obrazové elementy. Tvoří se zejména ve spodních rozích obrazu, a to pokud je senzor umístěn níže jak cca 25 cm nad zemí. Tato práce tak musí s tímto problémem počítat.



(a) IR kamera (b) RGB kamera

Obrázek 7: Falešné elementy - na obrázku IR kamery lze vidět několik objektů, které neexistují.

4.5.1 Provedená měření

Všechna měření přesnosti dat (vzdálenosti) jsou na přiloženém CD k dispozici ve formátu *.csv, který byl pro Microsoft Kinect SDK v této práci navržen. První řádek poskytuje informace o nasnímaném dokumentu ve formátu:

Zdrojový kód 2: První řádek ukládaného csv souboru

```
NEAR; [hodnota pro objekty, které jsou příliš blízko];
FAR; [hodnota pro objekty, které jsou příliš daleko];
UNKNOWN; [hodnota pro objekty, jejichž vzdálenost je senzorem
    nezměřitelná];
NAME; [název dokumentu]
```

Další řádky pak odpovídají jednotlivým řádkům naměřených hodnot v daném rozmezí. K tomu je přiložen snímek formátu *.png s vyobrazením surových dat hloubkové kamery (získaných později pomocí frameworku AForge a OpenKinect) a barevný obrázek typu *.jpg zobrazující aktuální scénu. Měření se týkají přesnosti senzoru, schopnosti interpretace objektů a sběru dalších podkladových dat k vytvoření co nejlepšího snímacího modelu s co nejpřesnějšími výsledky. Na konkrétní měření budou odkazovat příslušné kapitoly spolu s vyvozením příslušných závěrů. Pro vizualizaci souborů je určen program Kinect view (viz 6.1).

4.6 Přesnost

Přesnost výsledků senzoru ovlivňuje kromě samotných kamer i jeho stabilita. Ta nebyla u testovaného kusu zcela ideální a vzhledem k záruční době zařízení nebylo možné senzor hardwarově upravit. Problémem zde byl pohybový kloub pro náklon senzoru, v němž není zařízení zcela stabilní. Pokud robot jede po nerovném povrchu, prudce zrychlí, nebo nějakým jiným způsobem dojde k rychlému impulzivnímu pohybu, senzor se velmi snadno v kloubu pootočí. U bočních částí dochází na nerovném povrchu k výkyvům 1-2 mm do výšky, při prudkém rozjezdu se senzor nakloní i o 3-5 mm. Je tedy nutné v určitých časových intervalech kontrolovat požadovaný náklon senzoru. Senzor by měl být také pokud možno co nejvíce chráněn před nárazy a rychlými změnami pohybu, pokud nebude přinejmenším v ohybovém kloubu nějakým způsobem dodatečně fixován. Zároveň je vhodné zamezit možnosti protočení v kloubu pro případ, že hrozí náraz do jeho boční části - senzor se v takovém případě velmi snadno horizontálně pootočí.

Závěry diskuzních fór (př. stackoverflow) vedou k faktu, že hloubka měřená Kinectem není zcela přesná. Z tohoto důvodu byla vypracována série měření vzdálenosti v několika potenciálních výš-

kách, kde by mohl být senzor na platformu iRobot Create umístěn. Měření probíhala nasnímáním rovné stěny z určité vzdálenosti a výšky.

Jak je vidět na tabulce v příloze B, senzor trpí ve větších vzdálenostech i odchylkou 10 cm, při vzdálenostech do 2 m byla odchylka do 5 cm. Z toho plyne, že je senzor nejvhodnější pro měření do vzálenosti 2 m, dále výsledky nemusejí svou přesností dostačovat. Kompletní výsledky měření se nacházejí na přiloženém cd ve formátu uvedeném v předešlé kapitole.

Z šířky obrazu lze určit vzdálenost k boku předmětů. Pokud se vychází z dat, jež poskytuje Microsoft Kinect SDK, boční vzdálenost lze dopočítat jako: $\mathrm{Tg}(\alpha) \times$ vzdálenost bodu objektu, kde α je úhel od středu obrazu k danému předmětu. Tento úhel lze dopočítat jednoduchou přímou úměrností, pokud známe úhel, který tvoří celou šířku obrazu. Problematika tohoto údaje je rozebrána na konci předešlé kapitoly. Při nastavení 59° (obrázek 7) bylo dosaženo přesnosti s odchylkou přibližně 1-3 cm.

4.7 Rozpoznaní povrchů a objektů

Nevýhody IR senzorů jsou zmíněny již v kapitole 3.1.2. Hlavními podmínkami pro funkčnost tohoto řešení je odrazivá plocha. Nutné tedy bylo položit si otázku, jak si Kinect poradí v různých podmínkách.

Jako první byla testována odrazivost a funkčnost na různých typech podloží. Na následujících obrázcích lze vidět poměrně kvalitní výsledek na koberci, linoleu a dlaždicích. Oproti tomu leštěné parkety senzor výrazně matou a objekt na nich často není vůbec rozpoznán, popřípadě s velkou chybovostí. Při venkovním užití záleží na světelných podmínkách. Čím ostřejší slunce, tím bylo rozpoznání horší. Opět docházelo i k mizení objektů.

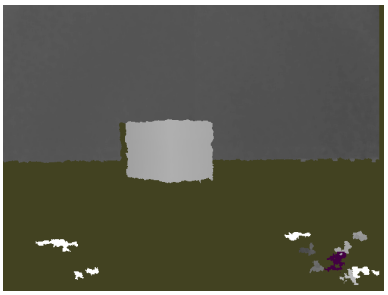
(a) Koberec RGB

(b) Linoleum RGB

(c) Dlaždice RGB

(d) Koberec IR

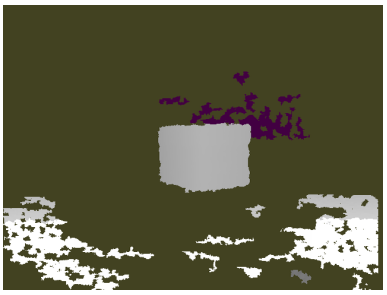
(e) Linoleum IR



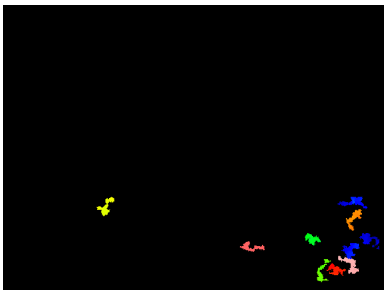
(f) Dlaždice IR

Obrázek 8: Test snímání Kinectu na různých podložích - bezproblémové prostředí.

(a) Parkety



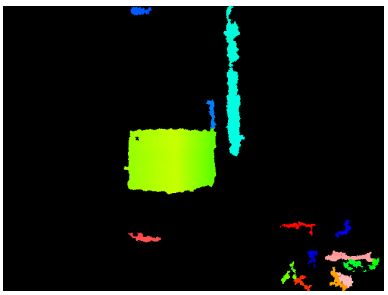
(b) Vysoký šum



(c) Mizející objekt

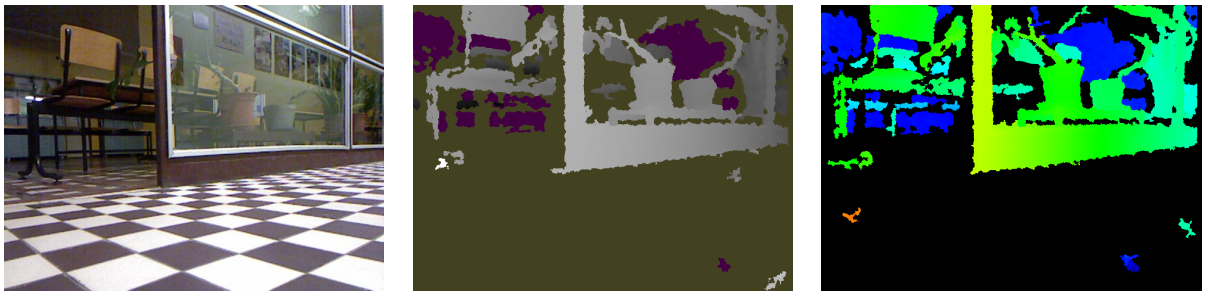
(d) Venkovní dlažba (beton)

(e) Mizející objekt



(f)

Obrázek 9: Test snímání Kinectu na různých podložích - problematické prostředí (snímky c a f pořízeny pomocí AForge.NET a OpenKinect).



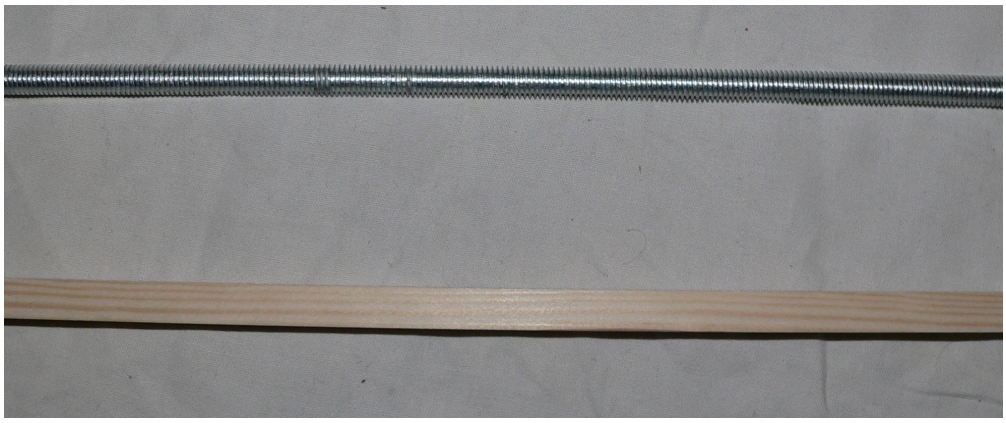
(a) RGB kamera (b) IR - Kinect SDK (c) IR - OpenKinect

Obrázek 10: Se stěnou si senzor poradí bez větších problémů, skleněnou překážku ale vůbec nevidí.

Dalším testem byl minimální interpretovatelný předmět. Jako příklad ideálního odrazivého materiálu posloužily dřevěné latě, jako příklad těžko rozpoznatelného objektu kovové tyče s vruty. Testovány byly následující průměry objektů: 0,5; 1;1,5 a 2 cm. Test probíhal ve dvou jejich polohách: a) položené na zemi; b) 10 cm vysoko v prostoru. Test dopadl s následujícími závěry:

Tabulka 2: Test měřitelných objektů - minimální měřitelný objekt

předmět	dřevěná lať		železná tyč	
pozice	na zemi	ve výšce 10 cm	na zemi	ve výšce 10 cm
vzdálenost v cm	průměr v cm a celistvost objektu v %			
100	0,5 - 100%	1 - 100% 0,5 - 5%	0,5 - 70% 1 - 90%	1,5 - 95% 1 - 40%
150	1 - 95% 0,5 - 40%	1,5 - 100% 1 - 70%	1 - 40% newline 1,5 95%	1,5 - 10% 2 - 80%
200	1,5 - 100% 1 - 95%	2 - 100% 1,5 - 50%	2 - 80% 1,5 - 60%	0%
250	1,5 - 95% 2 - 100%	1,5 - 30% 2 - 100%	2 - 20%	0%
300	2 - 90%	2 - 40%	0%	0%



(a) Ukázka materiálu použitého v testu.

(b) Kovová tyč s průměrem 1 cm ve vzdálenosti 100 cm.

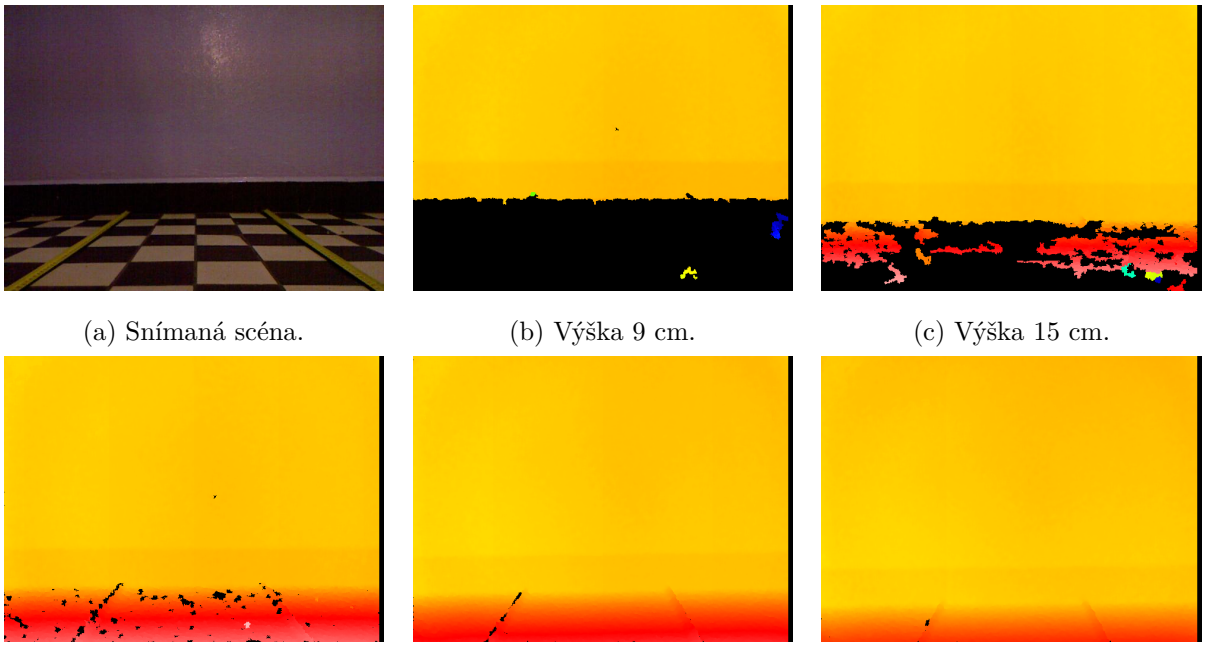
(c) Dřevěná lať s průměrem 1,5 cm ve vzdálenosti 150 cm.

(d) Dřevěná lať s průměrem 0,5 cm ve vzdálenosti 100 cm.

Obrázek 11: Ukázka z testu minimalních objektů, všechna data naleznete na přiloženém CD.

4.8 Výškové umístění a náklon senzoru

Během série měření byla zjištěna další podstatná informace. Pokud senzor umístíme dostatečně nízko, bude interpretovat podlahu jako neznámá data. Toho lze využít u vyhledávání objektů - scéna se zjednoduší odečtem nevalidních dat. Byla otestována umístění 9, 15, 20, 26 a 32 cm (kompletní data na přiloženém cd).



(a) Snímaná scéna. (b) Výška 9 cm. (c) Výška 15 cm.

(d) Výška 20 cm. (e) Výška 26 cm. (f) Výška 32 cm.

Obrázek 12: Ukázka z testu výškového umístění senzoru, uvedené fotografie jsou pořízeny ze vzdálenosti 100 cm (zaznamenáno pomocí AForge.NET a OpenKinect). Všechna data naleznete na přiloženém CD.

Z obrázků je patrné, že čím výše senzor leží, tím má obraz méně chyb. Výška 9 cm má jasnou přechodovou hranu, ale objevují se chyby v obraze. Výška 15 cm chyby v obraze obsahuje také, navíc trpí nejasnou přechodovou hranou. Detekce validních objektů je tedy těžší. Od výšky 20cm začínají rychle ubývat obrazové chyby a Kinect začíná korektně interpretovat i podlahu místnosti. Ve výšce 32 cm je obraz pouze s minimem chyb rozpoznání. Záměru mapovat objekty v místnosti, tedy ležící nebo vycházející z podlahy, vzhledem k uvedeným vlastnostem nejlépe vyhovuje výška umístění 9 cm.

Pokud umístíme senzor do výšky 9 cm, přibližně třetina obrazu tak zůstane nevyužita (interpretuje nevalidní data). Obraz se dá opětovně využít nakloněním senzoru o několik stupňů. Zde je nutné položit otázku, zda má tento krok pro projekt význam. Pokud budeme chtít zachytit celou osobu pro využití rozpoznávání gest, musí být od robota značně vzdálena. Aby robot stále viděl zem před sebou na 80 cm, je možný maximální náklon 8°. Při tomto náklonu je osoba výšky 195 cm zachycena až ze vzdálenosti 3m, senzor už v této pozici však začíná trpět větší odchylkou přesnosti (viz tabulka 5 v kapitole 4.6). Pokud necháme náklon na 0°, osoba uvedené výšky není kompletně zachycena ani z maximální operační vzdálenosti senzoru, tj. 4m. Pro tento projekt byl náklon

vyhodnocen jako nepotřebný, senzor je tedy nastaven na 0°.

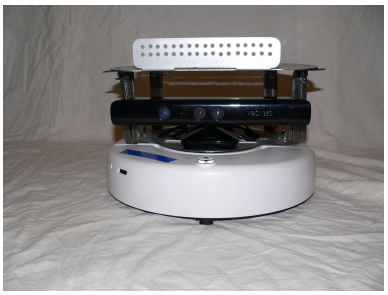
4.9 Osazení senzoru

Během bakalářského projektu byla pro platformu iRobot Create zhotovena kovová konstrukce, která byla připravena na namontování malých senzorů (předpokládalo se užití ultrazvukových nebo infračervených senzorů).

Tato konstrukce však není pro využití Kinectu vhodná. Nebylo zde možné provozovat senzor s otevřeným a přístupným počítačem, ani ho řádně uchytit. Proto byly navrženy a vytvořeny nové distanční sloupky spolu s podstavou pro Kinect. K ní je Kinect přichycen hmotou TACK-IT (hmota běžně využívaná k přichycení objektů, např. papírů k tabuli nebo jiné desce).

(a) Stará konstrukce.

(b) Nová konstrukce.



(c) Konstrukce se senzorem.

Obrázek 13: Konstrukce s novými distančními sloupky a podstavou pro senzor.

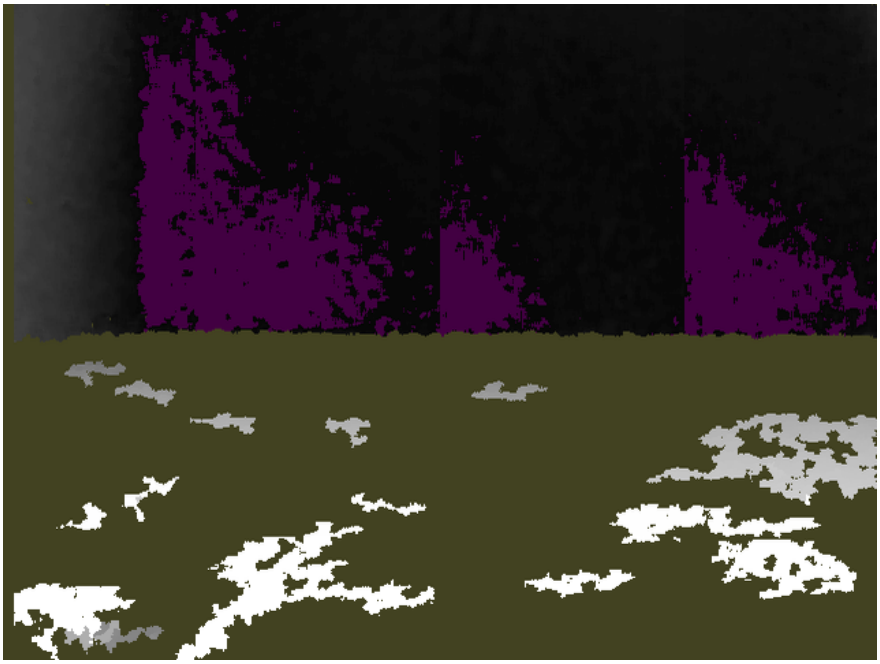
Celá konstrukce se tímto zvedla a je nyní výškově nastavitelná dle potřeby. Kinect se tak i s možným náklonem bez stínění kamer vejde pod ni. To umožní využití senzoru při minimálních koncepčních změnách a zároveň je možné využít nedokonalosti infračervené kamery pro zjednodušení měření (viz 4.8).

Problémem osazení Kinectu zůstalo napájení. Uvažované ultrazvukové a infračervené senzory bylo možné napájet 5V z USB portu počítače, popřípadě pomocí digitálních výstupů robota (taktéž 5V). Kinect oproti tomu potřebuje dodatečné napájení 12V. Bez přidání další baterie přicházelo v úvahu pouze vyvedení napětí z baterie robota, která má ale vyšší voltáž (14,4V). Baterie netbooku má voltáž nižší. Za pomoci elektrotechnika byl vybrán stabilizační obvod pro napájení Kinectu, nebyl ale prozatím realizován vzhledem k problémům napájení a nabíjení samotné platformy. Tyto problémy v současné době řeší doktorandi MTI TUL - bez jejich závěrů nebude obvod prozatím realizován. Za schéma a pomoc s výběrem obvodu děkuji Jakubu Štěpánkovi, naleznete jej v příloze C.

4.10 Microsoft Kinect jako senzor v mobilní robotice

Po několikaměsíčním testování bylo možné z práce se senzorem vyvodit následující závěry:

- Senzor ve své původní verzi pro Xbox 360 v kombinaci s Microsoft Kinect SDK je vhodným senzorem pro robotickou exploraci v rozmezí 80 cm až 2 m. Nová verze senzoru - Kinect pro Windows - umožňuje měření již od 40 cm, je tak pro pro vnitřní mobilní robotiku vhodnější. Toto omezení neplatí při využití ovladačů OpenKinect, které mají přímý přístup ke kameře senzoru.
- Senzor je od cca 2m poměrně nepřesný. (4.6)



Obrázek 14: Ukázka zpracování obrazu Microsoft Kinect SDK pro velkou vzdálenost. Pro tento snímek byl senzor vůči stěně nakloněn - bližší roh byl ve vzdálenosti 376 cm, vzdálenější pak 382 cm. Z obrázku je patrné zpracování obrazu v menších obdelnících.

- Senzor vyžaduje relativně výkonný hardware (4.4.4)
- Senzor korektně funguje pouze na určitém povrchu (4.7).
- Senzor je náchylný na světelné podmínky - při ostrém slunci snímání hloubky nefungovalo zcela korektně (mizení předmětů, šum - více 4.7).
- Dle diskuzních fór součinnost více Kinectů zavádí do výsledků nechtěný šum (vzájemné rušení). Možné řešení problémů nabízí jeden z projektů programu Microsoft Research [11].

V jazyce C lze pod distribucí Linuxu Kinect pro Xbox360 v mobilní robotice dále testovat s využitím ovladačů OpenKinect, volitelně s knihovnou OpenNI schopnou rozpoznat osoby, gesta atp. Zde by neměl být problém s omezením minimální snímatelné vzdálenosti, může ale opět nastat problém s nekompatibilním hardware pro vykreslování. V jazyce C# lze využít s ovladači OpenKinect knihovnu AForge.NET. Bohužel aktuálně nejsou dostupné informace o funkčnosti těchto řešení s novým Kinectem pro Windows.

Při využití senzoru ve verzi pro Xbox360 spolu s oficiálními ovladači Microsoftu je vhodné senzor využít pro úlohy, kde bude snímaní objektů probíhat ve vzdálenosti od 80 cm do 2 m.

5 Zpracování obrazu

Počítačové vidění je samostatný vědní obor skládající se z mnoha obsáhlých částí, od studia jednotlivých komponent optických soustav (čipů, objektivů, ...), přes jeho přenos, zpracování až k výslednému zobrazení. Tato věda sama o sobě dalece přesahuje rozsah této práce, a proto zde budou uvedeny pouze základní operace a pojmy, které byly při práci použity, testovány, nebo s ní úzce souvisí. Teoretická část kapitoly je zpracována na základě přednášek inženýra Horčičky [7].

Základním problémem u zpracování jakéhokoliv obrazu je informace o tom, co se vlastně v obrazu hledá. Úlohy zpracování obrazu se většinou danou posloupností operací specializují na určitý typ úkolu - např. rozpoznej v obraze kulatý objekt se zadaným průměrem a zjisti, zda jeho část nechybí. Aplikovat metodiku, která bude schopna naprosto univerzálně interpretovat veškerý obrazový materiál je vysoce náročná úloha sama o sobě, dalece přesahující rozsah této práce. Proto je v této práci navržen pouze základní algoritmus pro vyhledání spojitých objektů. Jejich zpracování dále závisí na užití v konkrétní úloze ke konkrétním účelům.

5.1 Segmentace obrazu

Segmentací obrazu se rozumí proces, kdy je obraz rozdělen do několika skupin (tj. segmentů) dle jejich určité podobnosti (totožné vlastnosti). Nejpoužívanějším způsobem segmentace je prahování.

5.1.1 Prahování

Základním parametrem prahování je práh. Ten může být určen staticky (pevně zadaný) nebo adaptivně (výsledek určité předchozí analýzy daného obrazu). Při prahování vznikne z jakéhokoliv obrazu (barevného, černobílého) obraz binární, tj. pole, kde hodnoty nabývají pouze hodnot 0 a 1 (případně true/false, dle interpretace). Nová hodnota se určí porovnáním prvku původního obrazu s prahem. Pokud je hodnota větší rovna prahu, je výsledný element zaznamenán jako 1 (true). Pokud je hodnota vůči prahu menší, výsledkem je 0 (false).

5.2 Matematická morfologie

Jedná se o soubor matematických transformací určených pro zpracování obrazových dat, jejichž postupy jsou založeny na množinových operacích. Dělí se na dvě základní kategorie - morfologii šedotónovou a morfologii binární. Při práci byly testovány dvě základní metody binární morfologie - binární dilatace a binární eroze - a dvě metody složené - binární otevření a binární uzavření.

Za ilustrační obrázky operací pro tuto kapitolu děkuji Petru Brantovi.

5.2.1 Binární dilatace a eroze

Jedná se o základní metody matematické binární morfologie. Obě operace pracují kromě samotného binárního obrazu se strukturním elementem. Strukturní element je zpravidla menší bodová množina než samotný reprezentovaný obraz. Dále je pro operace nutné znát pozici (tzv. reprezentativní bod), kterou se má daný element na obrazovou matici přikládat.

Binární dilatace

Binární dilatace zvětšuje objekty, zaplňuje malé díry a spojuje blízké objekty v binárním obraze. Principem je součet obou množin - binárního obrazu a strukturního elementu podle reprezentativního bodu. Operace je pro množiny komutativní a je reprezentována následujícím vztahem:

$$X \oplus B = \{p \in E^2 : p = x + b, x \in X \wedge b \in B\},$$

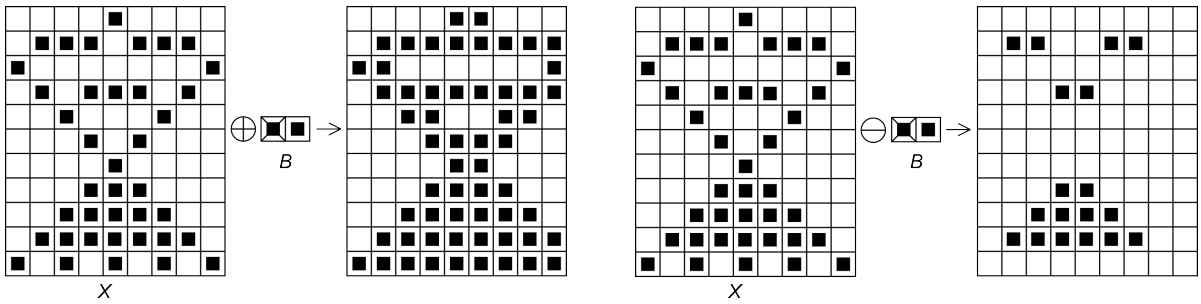
kde X je množina a B strukturní element (reprezentativní bod označen křížkem).

Binární eroze

Binární eroze zmenšuje objekty, ruší objekty menší než daný strukturní element a odděluje šíje (úzké spojnice objektů). Principem je porovnání shody obrazové matice se strukturním elementem. Operace je reprezentována následujícím vztahem:

$$X \ominus B = \{p \in E^2 : p = x + b \in X pro\ každé\ b \in B\},$$

kde X je množina a B strukturní element (reprezentativní bod označen křížkem).



(a) Binární dilatace (b) Binarní eroze

Obrázek 15: Základní metody binární morfologie.

5.2.2 Binární otevření a uzavření

Hlavním účelem složených metod je nezměnit velikost objektů při aplikování vlastností první operace (tj. dilatace nebo eroze). Operace jsou idempotentní - opakované provedení dále původní výsledek nemění.

Binární otevření

Jedná se o posloupnost operací eroze následovaná dilatací s využitím stejného strukturního elementu. Využívá se pro zjednodušení struktury, odstranění úzkých spojnic a malých objektů při zachování velikosti ostatních. Operace je reprezentována následujícím vztahem:

$$X \circ B = (X \ominus B) \oplus B,$$

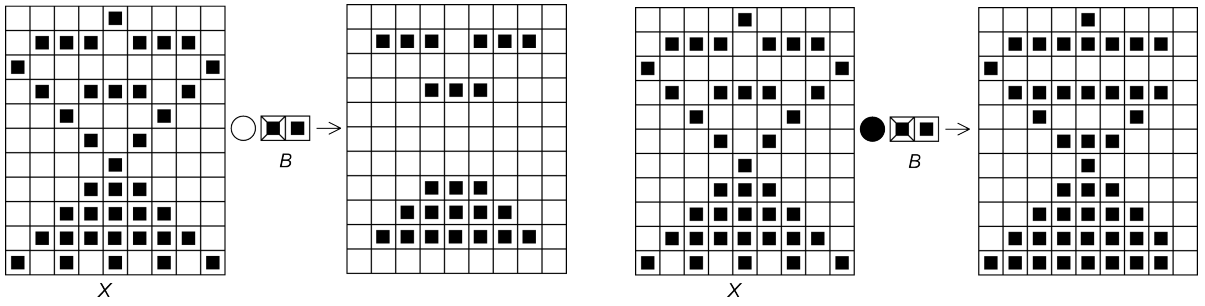
kde X je množina a B strukturní element (reprezentativní bod označen křížkem).

Binární uzavření

Opačná posloupnost vůči binárnímu otevření, tedy posloupnost dilatace následovaná eroze s využitím stejného strukturního elementu. Operace spojuje blízké objekty, zaplňuje malé díry při zachování velikosti ostatních objektů. Operace je reprezentována následujícím vztahem:

$$X \bullet B = (X \oplus B) \ominus B,$$

kde X je množina a B strukturní element (reprezentativní bod označen křížkem).



(a) Binární otevření

(b) Binarní uzavření

Obrázek 16: Složené metody binární morfologie.

5.3 Hranové detektory

Hrany jsou v barevném obraze místa, kde se prudce mění hodnota jasu. K jejich detekci se používá několik možných přístupů. Z možných řešení jmenujme: Fourierovu transformaci (2D), využití matematické morfologie a hranové operátory. V práci byly testovány postupy matematickou morfologií a hranové detektory.

5.3.1 Detekce hran matematickou morfologií

V této kategorii se nabízejí tři možná řešení:

- odečtení originálního obrazu od dilatovaného;

- odečtení erodovaného obrazu od originálního;
- odečtení erodovaného obrazu od dilatovaného.

Všechna řešení jsou založena na vytvoření dvou obrazů, kdy jeden bude mít objekty větší a druhý menší. Podle zvoleného typu jsou výsledkem vnější nebo vnitřní hrany objektů. Pokud odečteme originální obraz od dilatovaného, získáme vnější hrany objektů. Výhodou této kombinace je, že nedojde ke ztrátě informace, která se využívá u eroze. Další možností je odečtení erodovaného obrazu od originálního, tímto získáme vnitřní hrany objektů. Třetí možnou kombinací je odečtení erodovaného obrazu od dilatovaného. V tomto případě budou hrany objektů přesahovat na obě strany.

5.3.2 Detekce hran hranovými operátory

Tyto operátory jsou většinou založeny na hledání hran pomocí derivací. Vzhledem k faktu, že se u práce s digitálním obrazem jedná o diskrétní signál, jsou v tomto případě nahrazeny derivace diferencí.

Operátory jsou založeny na dvou základních metodách. Hledání maxim u první derivace (Robertsův operátor, Sobelův operátor) nebo průchod nulou u derivace druhé. Základními parametry (krom založení na diferenci) jsou velikost okolí a směrová závislost. Nová hodnota každého bodu se získá přiložením příslušné konvoluční masky k zadanému obrazu (reprezentační maticí hodnot). Zjednodušeně lze operaci popsat jako průměrování, kde mohou mít různé body okolí odlišnou váhu. Nejznámějšími operátory jsou:

Robertsův operátor

Pravděpodobně nejstarší a nejjednodušší operátor pracující s okolím 2x2.

Sobelův operátor

Základní okolí 3x3, může být i větší. Je směrově závislý, celkem 8 jader, pokud budeme hledat hrany ve všech směrech.

Laplaceův operátor

Směrově nezávislý, založený na druhé derivaci. Využívá se například v ostřících algoritmech. Základní hranový detektor pro knihovnu AForge.NET (třída Edges).

$$\left|\begin{matrix}1 & 0\\ 0 & -1\end{matrix}\right| \quad \left|\begin{matrix}0 & 1\\ -1 & 0\end{matrix}\right| \qquad \left|\begin{matrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{matrix}\right| \qquad \left|\begin{matrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{matrix}\right|$$

(a) (b) (c)

Obrázek 17: Ukázky operátorů: a) Robertsovi operátory; b) jedno z jader Sobelova operátoru; c) varianta Laplaceova operátoru (implementovaná v AForge.NET).

5.3.3 Výběr hranového detektoru

Požadavky na hranový detektor pro zadanou úlohu byly rychlost a přiměřená citlivost. Tyto podmínky vyřadily hranové operátory pracující ve více než dvou směrech s větším okolím.

Při testech se velmi dobře osvědčila základní směrová derivace v obou osách obrazu, tj hranové operátory:

$$\left|\begin{matrix}1\\ -1\end{matrix}\right| \qquad \left|\begin{matrix}1 & -1\end{matrix}\right|$$

Obrázek 18: Vybrany hranový operátor.

Výsledky byly natolik uspokojivé a pro daný úkol dostačující, že se toto jednoduché řešení s volitelným prahem využilo jako hlavní projektový hranový detektor.

5.4 Vyhledání objektů v hloubkovém obraze Microsoft Kinect

5.4.1 Návrh

Bylo potřeba vytvořit metodu, jež bude schopna rozpoznat z hloubkového obrazu Kinectu jednotlivé objekty. Metoda by měla být co nejuniverzálnější pro různé druhy a tvary objektů. Programátor by po aplikaci měl obdržet data interpretující jednotlivé objekty a jejich velikost.

Navrhované řešení bylo složeno z několika obrazových operací.

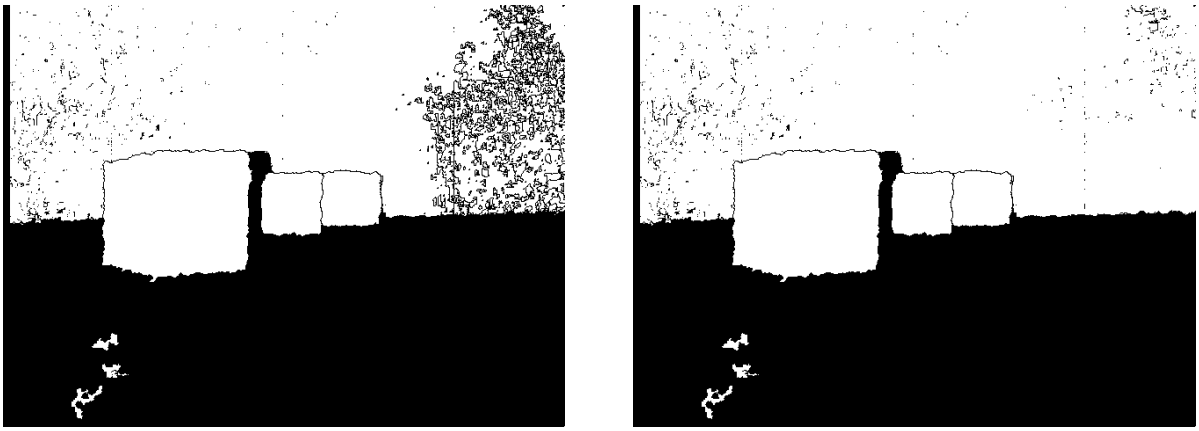
- Je nutné vytvořit hranový detektor s volitelnou citlivostí, který zaznamená jednotlivé hrany objektů a vytvoří z nich binární obraz.
- Vytvoří se binární obraz, který vznikne součtem obrazu z hranového detektoru spolu s obrazem nevalidních dat (tj. data, která Kinect označí za neznámá).

- Filtrace šumu v obraze hran.
- Na vzniklém binárním obraze proběhne vyhledání objektů, z kterého bude zaznamenán počet pixelů, které zabírá daný objekt. Dále by bylo vhodné vytvořit mapu, která bude zaznamenávat jednotlivé objekty seřazené podle velikosti.
- Výslednou mapu a data objektů pak lze libovolně využít pro nalezení nebo práci s jednotlivými objekty dle jejich velikosti a umístění.

5.4.2 Implementace

Popis vybraného hranového detektoru je k dispozici v kapitole číslo 5.3.3. K umožnění jeho aplikace bylo nutné vytvořit z obrazu dvě matice hodnot obsahující výsledky po průchodu hranových operandů. Z těchto matic se následně prahováním vytvoří binární obraz (prahování viz kapitola 5.1.1). Experimentovalo se s různou statickou hodnotou prahu - jako optimum mezi dostatečnou citlivostí a šumem byla zvolena hodnota 60.

K binárnímu obrazu hran se přičtou neidentifikovatelné oblasti, uložené při získávání dat. Pro redukci obrazových chyb je možné provést součet z několika po sobě jdoucích snímků.



(a) Nastavení 40 (b) Nastavení 60

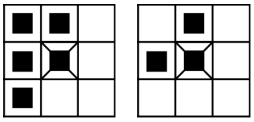
Obrázek 19: Výběr minimální prahové hodnoty - hodnota 60 má přijatelný šum při snaze o zachycení co nejvíce reálných hran.

Získaný obraz interpretuje okraje rozpoznaných předmětů. Bohužel prahování není dokonalé a většinou zanese do obrazu chyby ve formě zbytečných elementů, které jsou tvořeny několika body v prostoru. Před dalším postupem je vhodné obraz těchto oblastí zbavit. Erozi, se kterou se pro tento případ počítalo, použít nelze, protože dojde ke značné ztrátě potřebné informace. Proto je v tuto chvíli aplikována metoda pro rozpoznání objektů na získaný hranový obraz. Z výsledného

objektu se dál použijí pouze elementy, které jsou složeny z více než zadaného počtu bodů. Tím je zaručeno odfiltrování malých objektů při minimálně ztrátě užitečné informace. Zároveň je možné před tímto krokem použít binární dilataci pro zaplnění malých mezer mezi hranami, je ale třeba počítat s faktem, že se všechny objekty zvětší. Další možností je binární uzavření - při jeho aplikaci zůstanou objekty stejně velké, nicméně výpočetní náročnost oproti samotné dilataci je dvojnásobná.

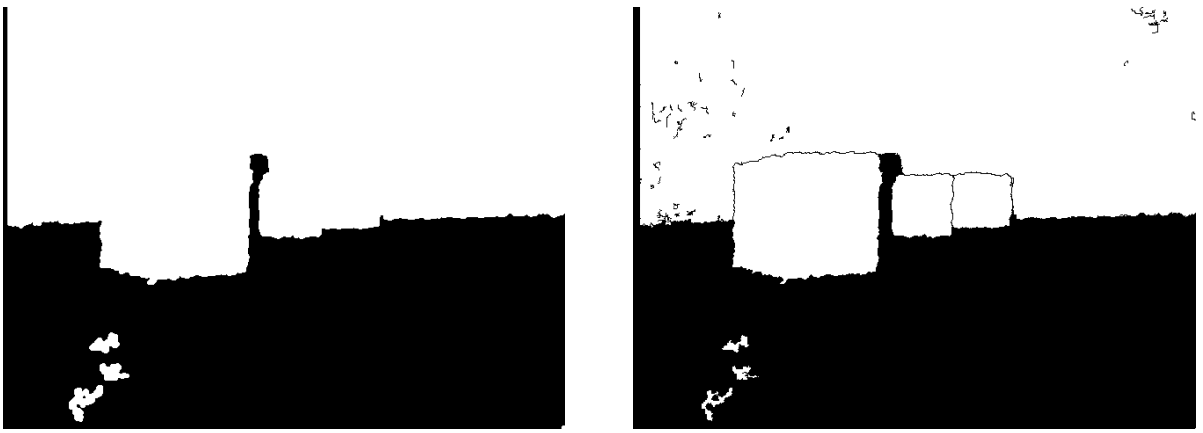
Samotný algoritmus použitý pro získání objektů z obrazu funguje následovně:

- Projdi matici po sloupcích, a pokud daný bod má být objektem (zjištěno dle vstupního binárního obrazu), zanes na pole do výsledné mapy číslo objektu. Číslo objektu se získá porovnáním následujícího okolí:



Obrázek 20: Testované okolí dle požadavku na diagonální propojení objektů.

- Pokud tato oblast už nějaký objekt obsahuje, přiřaď bod k danému objektu. Pokud obsahuje více objektů, zanes hodnoty do pole pro jejich následné spojení.
- Po průchodu celého obrazu spoj objekty dle uložené informace z předchozího kroku.
- Seřaď objekty od největšího k nejmenšímu a vytvoř mapu založenou na tomto indexu (tj. největší objekt s indexem odpovídající hodnotě 1, druhý největší hodnotě 2 atd.).



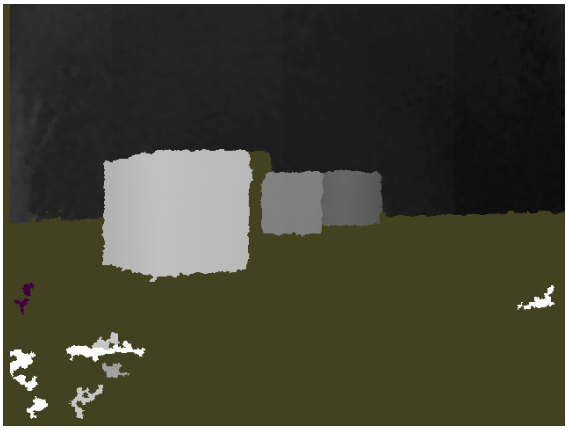
(a) Binární eroze (b) Zachování hranových objektů s obsahem >10

Obrázek 21: Redukce šumu - neúspěšné a úspěšné řešení. Jako vstup operací slouží snímek b z obrázku 19.

Výsledkem je objekt, ve kterém je uložena mapa spolu se seznamem bodů pro každý objekt.

Pro další práci se využije nový binární obraz vytvořený prahováním mapy z předchozího bodu. Dle počtu bodů pro každý objekt, které jsou v objektu uloženy, lze zvolit minimální požadovaný počet bodů. Pro testování byla použita hodnota 10.

Binární obraz hran má nyní redukovaný šum. V tomto kroku lze opět využít binární dilataci nebo binární uzavření pro zaplnění drobných mezer mezi hranami. Nyní se znovu aplikuje metoda na rozpoznání objektů, s tím rozdílem, že hranový obraz slouží jako oddělovač objektů. V tomto kroku se získá finální objekt obsahující mapu jednotlivých objektů a jejich velikost. Další zpracování získaných dat je volbou uživatele.



(a) Interpretovaná hloubková data

(b) Nalezené objekty (4 největší)

Obrázek 22: Největší nalezené objekty v obraze (jednotlivé barevně odlišeny).

Metoda je úspěšně aplikována v prezentační úloze „Explorace Kinect“, kterou naleznete na přiloženém cd. Konkrétně je umístěna ve třídě KinectDepth pod názvem findObjects vracející datový objekt objElements.

6 Vytvořené programy a knihovny

Tato kapitola obsahuje úlohy, jež byly k vytvoření práce nutné, a demonstrační úlohu robotické explorace s využitím zakoupeného senzoru Microsoft Kinect (kapitola 4). Celkem jsou popsány tři úlohy a jedna knihovna pro zařazení do dalších úloh se senzorem, která zahrnuje třídu pro získání obrazových dat z Kinectu a třídu s naprogramovanými operacemi pro zpracování hloubkového obrazu.

Úlohy jsou vytvořeny pro využití s Microsoft Kinect SDK v prostředí Microsoft Visual Studio 2010 pod .NET Framework 4. Všechny zdrojové kódy v jazyce C# naleznete na přiloženém CD. V příloze D naleznete snímky úvodních obrazovek všech aplikací.

6.1 Kinect view

Kinect view byla první nezbytná úloha ke splnění práce. Bylo nutné napsat program, který bude schopen získat obrazová data senzoru. Další úlohy pak mohou snadno implementovat použitý kód a naložit s daty podle potřeby.

6.1.1 Návrh

Bylo třeba vytvořit program, který demonstruje možnosti Kinectu využitelné v mobilní robotice. Program se tak zaměří na získání obrazových dat obou kamer a jejich nastavení, ovládání náklonu senzoru a získání úhlu mluvčího.

Sledování kostry bylo pro projekt a mobilní robotiku vyhodnoceno jako nepodstatné vzhledem k relativně velké vzdálenosti, ve které musí osoba vůči senzoru stát (podrobněji v kapitole 4.8).

6.1.2 Implementace

Aplikace byla úspěšně vytvořena dle dostupné dokumentace a je ve své poslední verzi funkční pod aktuálními ovladači (verze 1.0). Aplikace zpřístupňuje obraz obou kamer, umožňuje nastavit jejich rozlišení, dovoluje ovládat náklon senzoru, zobrazuje úhel mluvčího a navíc obsahuje možnost uložit právě zobrazený hloubkový snímek do formátu *.csv (podrobnosti o uložení dat naleznete v kapitole 4.5.1).

Při vytváření úlohy bylo zjištěno několik omezení pro zpracování zvuku přes senzor. Část, která se v objektu KinectSenzor stará o ovládání audia, má velkou časovou náročnost pro obnovu spojení po výpadku elektrické energie. Další problémy mohou způsobit známé chyby Microsoft Kinect SDK, a to konkrétně zrušení registrovaného audio streamu v případě následné aktivace vyhledávání

kostry (kompletní seznam známých chyb naleznete na stránkách Microsoftu věnovaných Kinect SDK). Po dodatečném zjištění, že senzor dokáže číselně interpretovat úhel příchozího zvuku pouze v rozmezí -50° až 50° před senzorem, se od dodatečného zpracování zvuku v této práci upustilo.

6.2 Kinect csv view

Motivací pro tuto úlohu se staly vysoké hardwarové požadavky senzoru. Zpočátku nebylo možné aplikovat testované metody pro zpracování obrazu přímo v reálném čase. Samotná režie senzoru s příjmem obrazu vytížila mobilní procesor Intel P8400 na cca 50%. Proto bylo do aplikace Kinect view implementováno uložení snímků do formátu *.csv, s kterými tento program pracuje (podrobnosti o formátu viz kapitola 4.5.1).

6.2.1 Návrh

Je třeba vytvořit program, který bude interpretovat dříve uložené snímky ze senzoru Kinect a umožní náhled testovaných operací s obrazem. Program bude výchozím bodem pro třídu starající se o zpracování obrazu.

6.2.2 Implementace

První verze aplikace se starala pouze o načtení uloženého snímku ve formátu *.csv. Postupně byly přidávány náhledy pro testované obrazové operace. Nyní aplikace zobrazuje:

- původní obraz,
- binární eroze[2] (viz 5.2.1),
- binární dilatace[2] (viz 5.2.1),
- binární otevření[2] (viz 5.2.2),
- binární uzavření[2] (viz 5.2.2),
- 1. derivaci (diferenci) s volitelným prahem,
- 2. derivaci (diferenci) s volitelným prahem,
- nalezení hranic pomocí metody vyhledávání objektů (viz 5.4),
- nalezení objektů pomocí metody vyhledávání objektů (viz 5.4),
- zobrazení bodů pro výpočet vektorů objektu (více motivace kapitola 6.4).

Vnitřní metody byly exportovány do projektu Kinect Driver, podrobnější následující kapitola.

[2]Strukturní element tvoří matice hodnot true o velikosti 5x5, v uživatelském prostředí není implementována volitelnost elementu.

6.3 Kinect driver

Kinect driver je *.dll knihovna kombinující funkce ovladače zařízení a metod zpracování obrazu. Kombinuje užitečné dosažené výsledky a umožňuje tak jejich využití v dalších projektech s využitím senzoru Kinect.

6.3.1 Návrh

Knihovna se rozdělí na dvě části. První bude zajišťovat získání dat ze senzoru a základní možná nastavení, druhá bude obsahovat metody pro zpracování obrazu hloubkové kamery.

6.3.2 Implementace

Knihovna obsahuje následující dvě třídy:

Kinect driver

Kinect driver je třída starající se o komunikaci a získávání dat ze zařízení. Umožňuje získat barevný obraz, spravovat náklon senzoru a získat zvolená data o hloubkovém obraze, na jejichž zpracování cílí. K dispozici jsou v tomto případě následující položky:

- matice surových dat tak, jak je vrátí Kinect (datový typ short),
- matice reálné hloubky předmětů (datový typ int),
- matice obsahující pozice osob (datový typ byte),
- matice předmětů, které jsou příliš blízko (datový typ bool),
- matice předmětů, které jsou příliš daleko (datový typ bool),
- matice hodnot, pro které nejsou data reálné hloubky, tj. nevalidní hodnoty (datový typ bool),
- bitmapa pro rychlé vizuální zobrazení dat, která senzor přijímá (datový typ Bitmap).

Konstruktor zahrnuje i dotaz na aktivaci sledování kostry a audia, vnitřní implementace je ale v tomto případě na programátorovi.

Kinect depth

Třída vytvořená z metod pro zpracování hloubkového obrazu původně obsažených v programu Kinect csv view. Třída obsahuje metody pro:

- získání pole první derivace (diference) ve směru x (5.3.3),
- získání pole první derivace (diference) ve směru y (5.3.3),
- získání pole libovolné vyšší diference,
- binarizaci obrazu prahováním (viz 5.1.1),

- binární dilataci (viz 5.2.1),
- binární erozi (viz 5.2.1),
- binární otevření (viz 5.2.2),
- binární uzavření (viz 5.2.2),
- metodu pro nalezení objektů v obraze (viz 5.4),
- metodu pro nalezení bodů pro vektor (6.4).

Dále třída obsahuje objekty pro funkčnost řešení.

6.4 Explorace Kinect

Explorace Kinect je cílová aplikace projektu, která má za úkol demonstrovat možnost využití Kinectu spolu s platformou iRobot Create v mobilní robotice. Aplikace využívá nabyté poznatky získané z předchozí analýzy senzoru a zpracování obrazu (kapitoly 4 a 5). K ovládání robota posloužila upravená knihovna IRobot Core vytvořená během bakalářského projektu. Snímek výřezu mapy naleznete stejně jako snímek samotné aplikace v příloze D.

6.4.1 Návrh

Bylo třeba vytvořit aplikaci, která dokáže zaznamenat překážky a bude ovládat robota tak, aby se jim dokazal vyhnout. Překážky budou vykresleny spolu s pozicí robota do mapy, která bude zobrazena uživateli.

K tomu je třeba zvládnout ovládání senzoru Kinect a základní zpracování obrazu. Těmto problémům se věnují kapitoly 4 a 5. Ovládání robota bude probíhat přes upravenou knihovnu IRobot Core s využitím ovládacích skriptů, které robot nabízí - ty jsou využity kvůli záměru o co největší přesnost robota.

Po analýze nedostatků senzoru byla zvolena jako vhodná prezentační úloha průjezd mezi krabicemi (konkrétně krabice od papíru, přibližná velikost 31,8 x 22 x 24 cm).

6.4.2 Implementace

Robot využívá ovladače zařízení z knihovny Kinect driver pro získání dat o reálné hloubce předmětů a mapy nevalidních údajů (tj. neznámá data, příliš blízké a příliš vzdálené objekty).

Vyhledávání objektů probíhá dle vytvořeného algoritmu pro rozpoznání objektů popsaného v kapitole 5.4, tj. aplikuje vytvořený hranový detektor, zbinarizuje obraz dle určeného prahu (použita hodnota 60), odstraní z obrazu malé objekty (menší než 10 bodů), výslednou mapu opět zbinarizuje

a nalezne v ní jednotlivé objekty. Aby bylo možné zakreslit objekty do uživatelské 2D mapy, byla třída Kinect depth v knihovně Kinect driver doplněna o následující metodu:

- Projdi zadaný obraz po sloupcích odspodu a zaznamenej pro každý objekt nejvýše 10 hodnot umístěných nejníže v každém sloupci.
- Vytvoř výsledný list objektů, kde každý objekt v listu reprezentuje jeden předmět v obraze. V daném objektu bude uložena zprůměrovaná hodnota reálné vzdálenosti z maximálně deseti bodů předmětu pro každý sloupec spolu s polohou nejníže položených bodů.

Metodu lze vyhledat ve třídě Kinect depth pod názvem findVectors.

Ze získaných dat proběhne analýza, zda je objekt (tj. krabice) vůči senzoru postaven jedním bokem, nebo v šikmém náklonu. Získané body se zakreslí do mapy vůči poloze robota spolu s oblastí (tzv. frontierem), kterou by svým středem neměl překročit, aby nedošlo ke kolizi. Dále se z uložených bodů zjistí, na které straně se objekt nachází - přesněji, na které straně od objektu má robot více prostoru. Tato data vytvoří pohybový skript pro robota, který probíhá ve třech krocích. Natočení do úhlu potřebného pro minutí překážky, jízda vpřed mírně za překážku, natočení robota do výchozího úhlu. Základní nastavení zadává robotovi, aby přejel vzdálenost nalezenou k hraně o 10 cm (účelem je minout objekt a nenarazit do něj v dalším kroku).

Aplikace vyhledává objekty dle velikosti, kdy má uživatel možnost požadované rozmezí sám definovat. Robot se řídí dle největšího předmětu, který v tomto rozmezí najde. Aplikace sčítá několik obrazů nevalidních dat (kapitola 4.5) za účelem snížení počtu falešných elementů v obraze. Pracuje s výřezem původního obrazu ve velikosti 540x300px s počátkem na souřadnicích 50x,50y z původního obrazu. Tento krok opět omezuje výskyt falešných elementů spolu se snížením výpočetní náročnosti. Uživatel kromě samotné mapy vidí náhled pozice aktuálně nalezeného objektu spolu s jeho spodní hranou. Tato vizuální data lze využít pro kalibraci nastavení. Program pracuje v celých pohybových sekvencích - tj. najdi požadovaný objekt, zakresli jej do mapy spolu s frontierem, proveď posun robota na novou pozici. Jejich start určuje uživatel. Video s touto úlohou se nachází na přiloženém CD, stejně jako zdrojový kód.

7 Závěr

V bakalářské práci byly úspěšně splněny všechny body zadání. Po stručném představení platformy byl zdůvodněn výběr dálkoměru vhodného k rozšíření schopností robota pro jeho větší autonomii. Následuje podrobný rozbor zakoupeného senzoru Microsoft Kinect (verze pro Xbox 360). Je představena technologie, na jejíž bázi senzor funguje, byly otestovány a rozebrány výhody a nevýhody aktuálně dostupných ovladačů zařízení. Následuje rozbor přesnosti výsledků, které senzor poskytuje. Konkrétně je věnována pozornost prostředím, kde může senzor fungovat, nejmenším objektům které senzor zaznamená a samozřejmě samotné naměřené vzdálenosti porovnané s reálnou hodnotou. Pozornost je věnována i umístění senzoru. Byla vytvořena nová část konstrukce pro možnost využití senzoru na platformě iRobot Create, dále tři aplikace a jedna knihovna pro využití senzoru v dalších projektech. Konkrétně se jedná o následující programy a knihovnu: aplikaci pro základní demonstraci možností senzoru, aplikaci pro ukázku různých operací na zpracování hloubkového obrazu senzoru, aplikaci demonstrující využití senzoru jako prostředek mobilní robotiky a knihovnu v jazyce C# usnadňující nasazení tohoto senzoru pro práci na dalších projektech v mobilní robotice. Tam může sloužit jako ovladač zařízení a základní prostředek pro zpracování dat senzoru. Práce dále obsahuje část věnovanou základním operacím zpracování obrazu, která obsahuje informace nutné k pochopení navrženého postupu explorace.

Na základě provedených měření z práce vyplývá, že senzor Microsoft Kinect ve verzi pro Xbox 360 spolu s oficiálními ovladači distribuovanými v Microsoft Kinect SDK je vhodným senzorem pro vnitřní exploraci pro vzdálenost od 80 cm do 2 m, kde není vyžadována 100% přesnost výsledků, ale je potřeba rychle prozkoumat obsáhlou scénu. Na projektu by bylo vhodné pokračovat s novou verzí Kinectu (Kinect pro Windows), který může s ovladači výrobce snímat předměty již od 40cm, což je pro mobilní robotiku výrazně lepší. Tomuto omezení se lze u stávajícího senzoru (ve verzi pro Xbox360) vyhnout nasazením jiných ovladačů, které poskytnou přístup přímo ke kameře senzoru. Výhody a nevýhody jednotlivých řešení jsou v práci na základě vlastních poznatků podrobně rozebrány.

Všechny postupy použité v práci jsou snadno implementovatelné do dalších projektů v jazyce C#. V budoucnu je rozšíření projektu možné přidáním dalších operací s hloubkovým obrazem, optimalizací stávajících a využití již vytvořených metod pro další libovolné úlohy. Vhodným příkladem může být detekce předmětů podle tvaru a ne pouze dle jejich velikosti a umístění. Dalším krokem projektu může být reprezentace okolního prostředí ve 3D, inspirací zde může být projekt KinectFusion.

Samotný senzor ve verzi pro Xbox360 má ohromný potenciál jako ovladač určitého softwaru pouze pomocí gest. V kombinaci s rozpoznáním hlasu může sloužit pro intuitivní ovládání libovolného programu bez nutnosti dalších periferních zařízení - to je také jeho původní účel a ten plní výborně. Další projekty s tímto konkrétním senzorem je tak vhodné směřovat do virtuálního ovládání a v mobilní robotice pokračovat ve vývoji se zmíněnou verzí senzoru pro Windows.

Literatura

[1] AForge.NET: AForge.NET Framework [online]. http://www.aforgenet.com/framework/docs/, 2012, [cit. 2012-05-01].

[2] Channel 9: Kinect for Windows SDK Beta 2 Quickstarts [online]. http://channel9.msdn.com/Series/KinectSDKQuickstarts, 2012, [cit. 2012-05-01].

[3] Code Laboratory: CL NUI Platform [online]. http://codelaboratories.com/nui/, 2010, [cit. 2012-05-01].

[4] Dan Krusi: Emss Framework - iRobot Create C++ Framework [online]. http://emssframework.sourceforge.net/emssframework_Main_Page, 2009, [cit. 2012-05-01].

[5] IRobot Corporation: IRobot Create Open Interface [online]. http://chess.eecs.berkeley.edu/eecs124/iRobotDocs/CreateOpenInterface_v2.pdf, 2006, [cit. 2012-05-01].

[6] IRobot Corporation: IRobot Create Owner's Guide [online]. http://www.irobot.com/filelibrary/create/Create/20Manual_Final.pdf, 2006, [cit. 2012-05-01].

[7] Jiří Horčička: Zobrazování obrazových dat [online]. http://elearning.fm.tul.cz/course/view.php?id=202, 2011, [cit. 2012-05-01] [dostupné po přihlašení].

[8] Micfosoft: Kinect for Windows [online]. http://www.microsoft.com/en-us/kinectforwindows/, 2012, [cit. 2012-05-01].

[9] Micfosoft: MSDN Academic Alliance Program [online]. http://www.microsoft.com/cze/education/licence/msdn_academic_alliance/default.mspx, 2012, [cit. 2012-05-01].

[10] Microsoft Research: KinectFusion [online]. http://research.microsoft.com/apps/video/default.aspx?id=152815, 2011, [cit. 2012-05-01].

[11] Microsoft Research: Shake 'n' Sense [online]. http://www.youtube.com/watch?v=CSBDY0RuhS4, 2012, [cit. 2012-05-01].

[12] Nagel, C.; Evjen, B.: *C# 2008 - Programujeme profesionálně*. Brno: Computer Press, 2009, ISBN 978-80-251-2401-7.

[13] OpenKinect community: OpenKinect project [online]. http://openkinect.org/wiki/Main_Page, 2012, [cit. 2012-05-01].

[14] Robot Electronics: Home to Advanced Sensors and Controllers [online]. `http://www.robot-electronics.co.uk/`, 2012, [cit. 2012-05-01].

[15] Sharp, J.: *Microsoft Visual C# Krok za krokem*. Brno: Computer Press, 2010, ISBN 978-80-251-31473.

[16] Stack Overflow: Precision of the kinect depth camera [online]. `http://stackoverflow.com/questions/7696436/precision-of-the-kinect-depth-camera`, 2011, [cit. 2012-05-01].

[17] Vvv : The Kinect Thread [online]. `http://vvvv.org/forum/the-kinect-thread`, 2010, [cit. 2012-05-01].

A Nejdůležitější parametry dálkoměrů

Tabulka 3: Parametry ultrazvukových senzorů

SFR02	**SFR08**	**SFR10**
500 - 550kč za jednotku	1300-1400kč za jednotku	1300-1400kč za jednotku
sériová linka nebo sběrnice I2C	připojení po sběrnici I2C	připojení po sběrnici I2C
možnost připojení až 16 senzorů na jednu sběrnici	možnost připojení až 16 senzorů na jednu sběrnici	možnost připojení až 16 senzorů na jednu sběrnici
dosah 16cm - 6m	dosah 3cm - 6m	dosah 3cm - 6m
vyzařovací úhel 55°	vyzařovací úhel 55°	vyzařovací úhel 72°
automatická kalibrace výkonu	ovladatelný výkon	ovladatelný výkon
jednotlivá čtení po 70ms	čtení v základu po 65ms - měnitelné v závislosti na výkonu	
	možnost detekce více objektů v různých vzdálenostech	
	světelný senzor	

Tabulka 4: Parametry IR senzorů

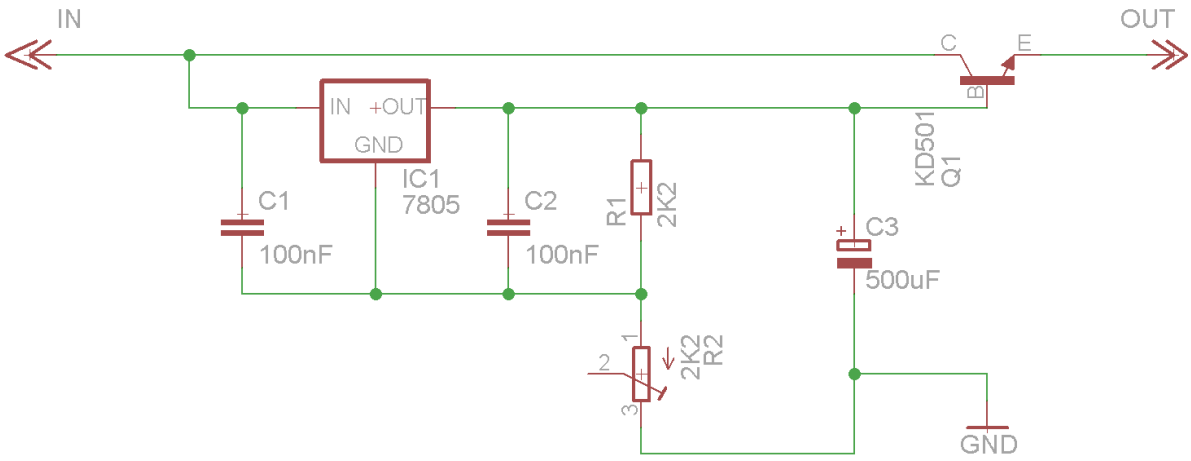
GP2Y0A02	**GP2Y0A21**
700 - 800kč za jednotku	300 - 400kč za jednotku
dosah 20 - 150cm	dosah 10 - 80cm
napájení 5V/33mA	napájení 5V/33mA
výstupem je napětí úměrné vzdálenosti	výstupem je napětí úměrné vzdálenosti

B Přesnost senzoru Kinect

Tabulka 5: Test vzdálenosti z výšky 15cm

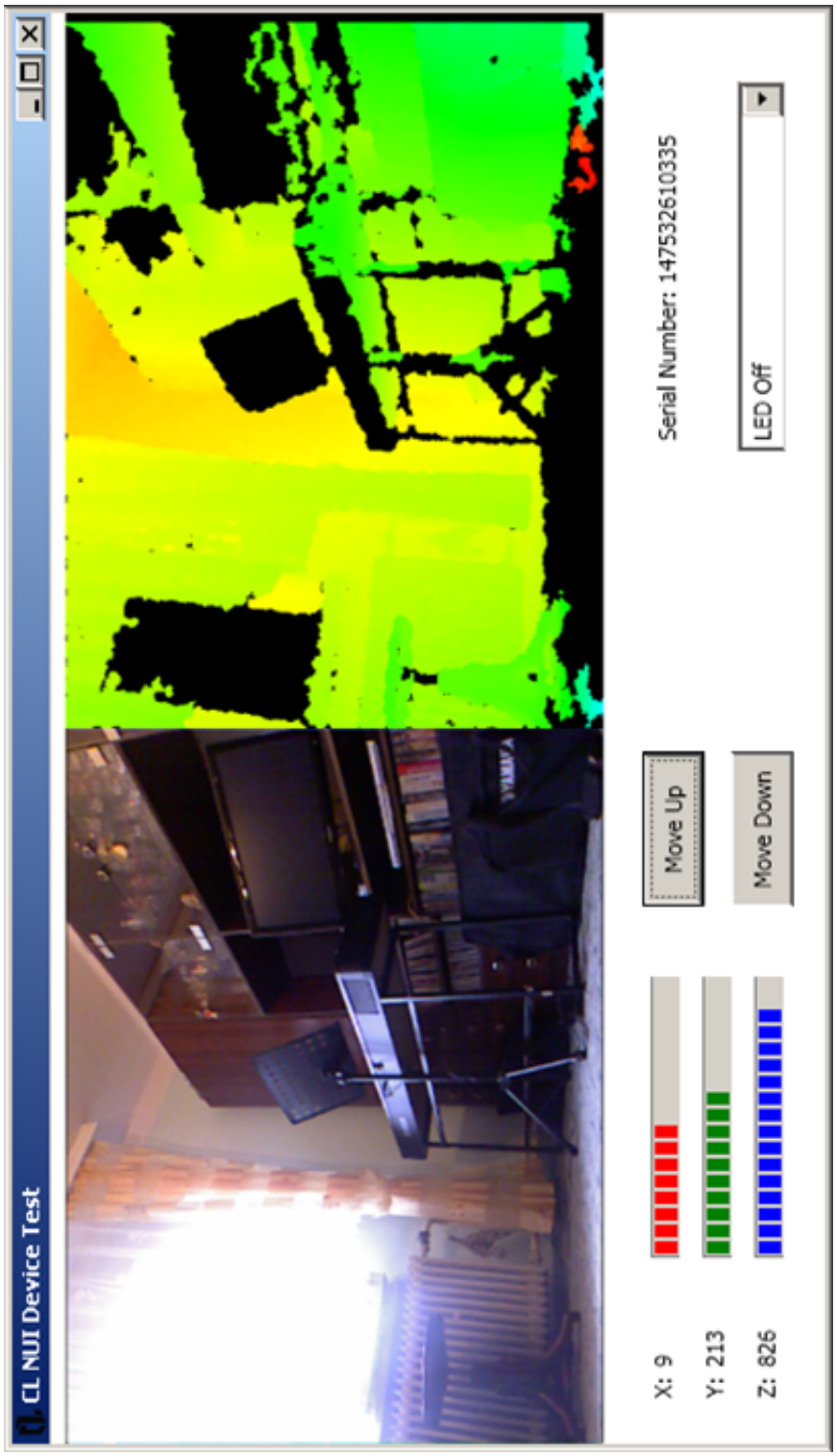
vzdálenost v cm	naměřená hodnota ve středu obrazu (vzdálenost v cm)	vzdálenost v cm	naměřená hodnota ve středu obrazu (vzdálenost v cm)
80	82,2	240	248,7
90	91,6	250	256,0
100	102,5	260	270,1
110	112,5	270	276,5
120	122,5	280	288,1
130	133,4	290	298,0
140	143,4	300	308,7
150	154,2	310	323,1
160	163,0	320	335,7
170	173,6	330	342,3
180	183,8	340	352,8
190	195,2	350	356,5
200	206,9	360	367,9
210	217,4	370	375,9
220	227,4	380	388,5
230	235,1	390	příliš daleko

C Stabilizační obvod

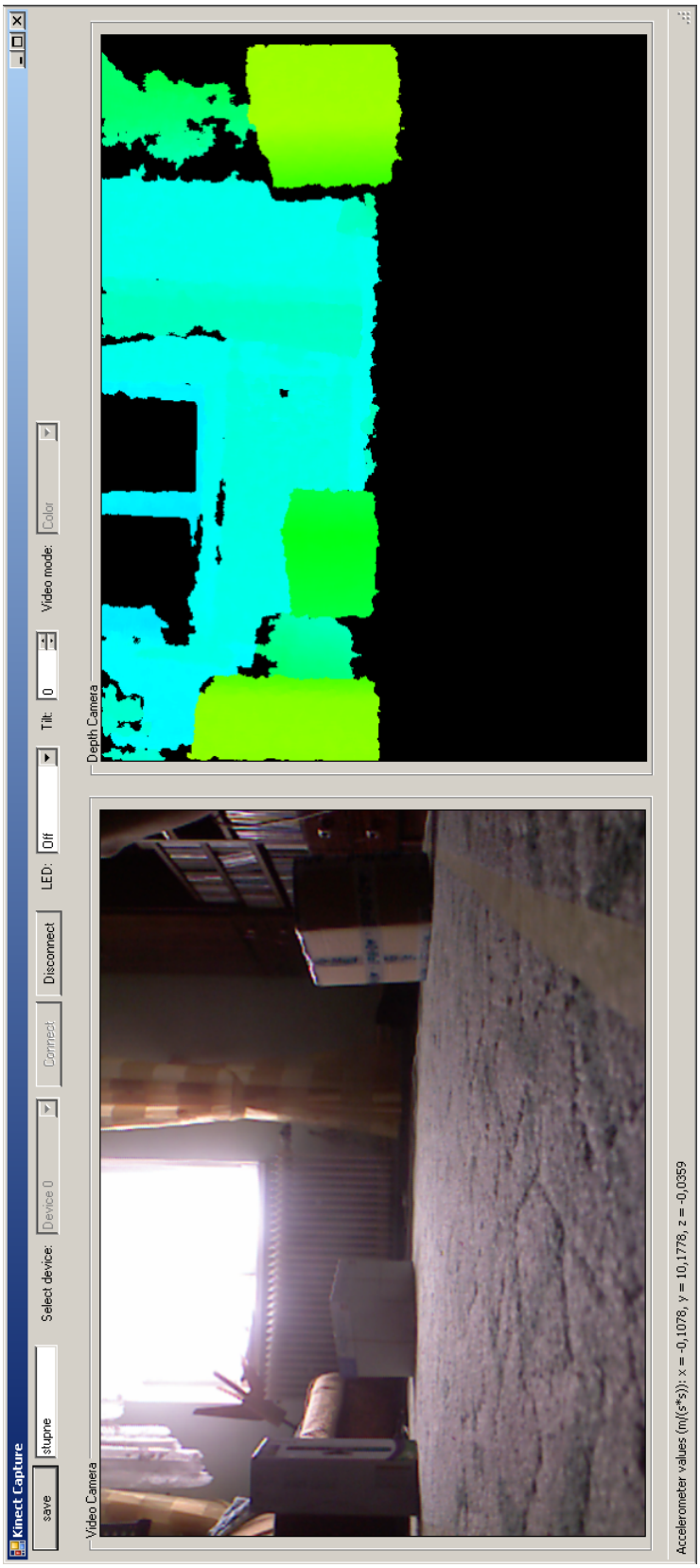


Obrázek 23: Možná varianta stabilizačního obvodu pro napájení senzoru Kinect z baterie robota.

D Ukázky z aplikací

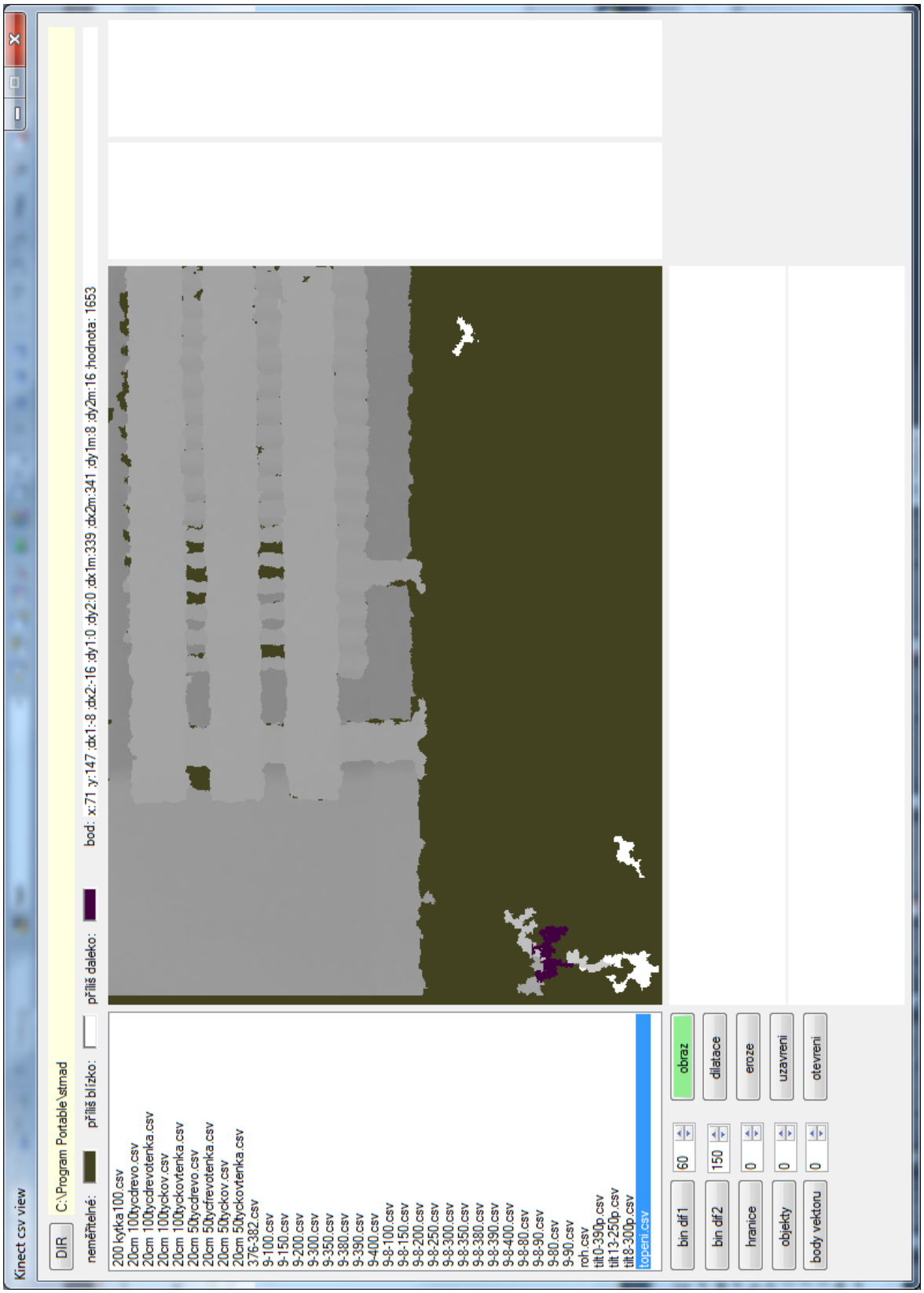


Obrázek 24: Testovací aplikace ovladaču CL NUI Platform.

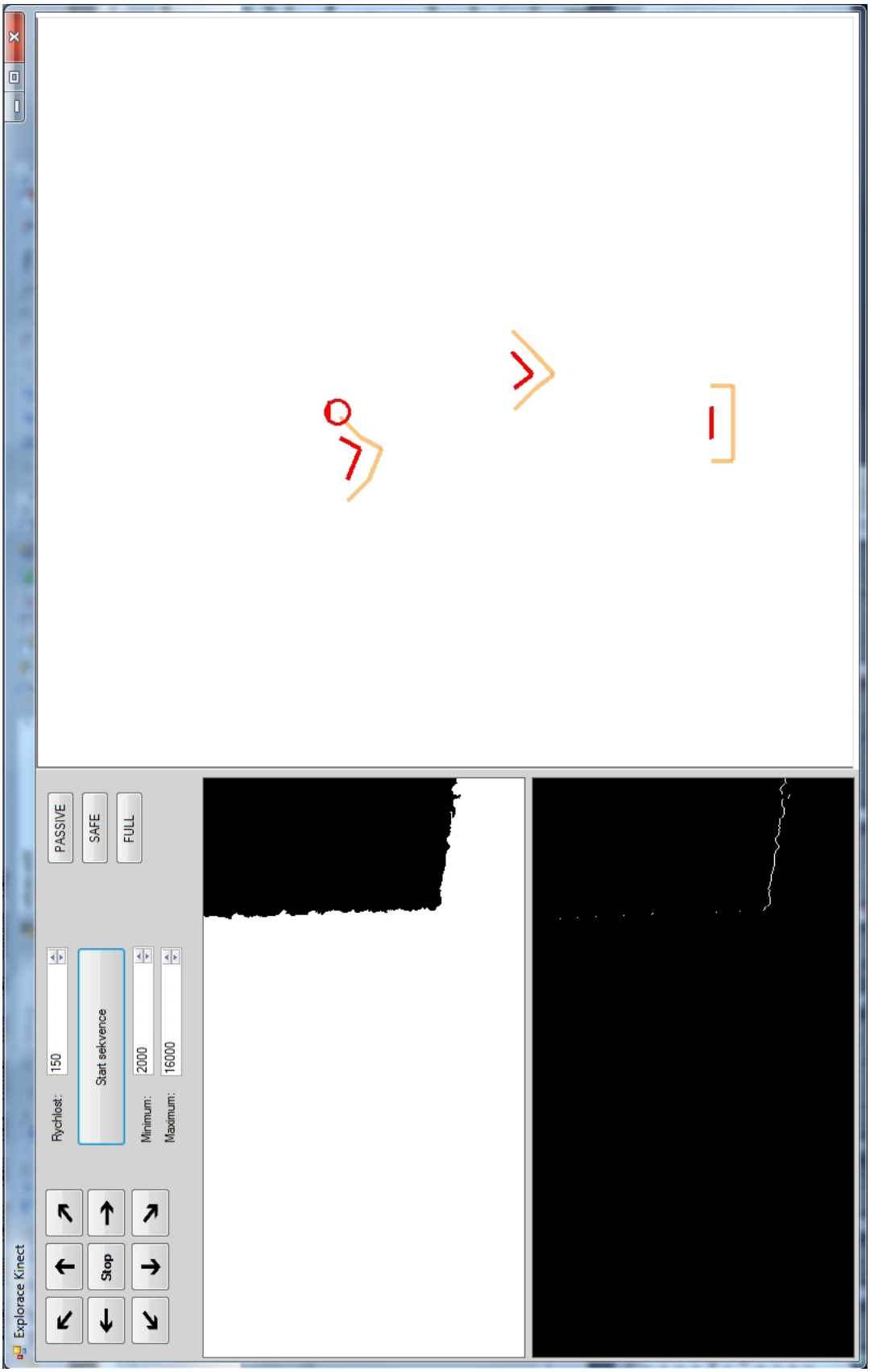


Obrázek 25: Testovací aplikace AForge.NET, založená na ovladačích OpenKinect.

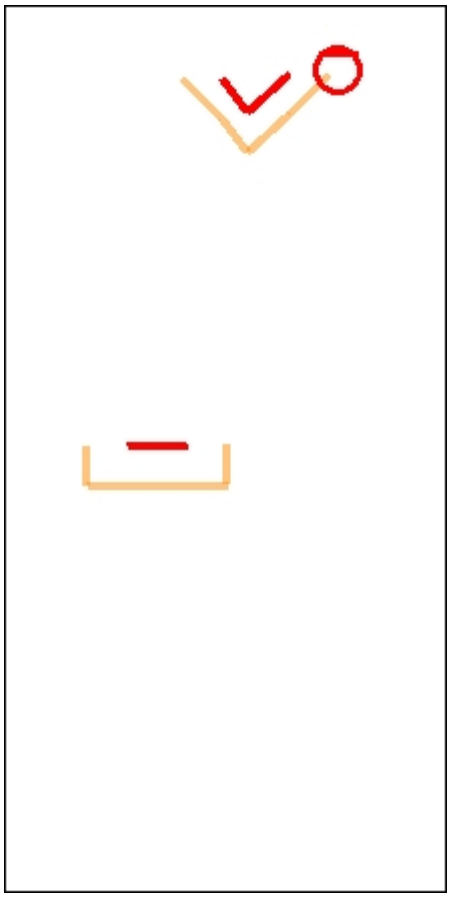
Obrázek 26: Testovací aplikace Microsoft Kinect SDK.



Obrázek 27: Aplikace Kinect csv view.



Obrázek 28: Aplikace Explorace Kinect.



Obrázek 29: Ukázka výsledné mapy, jeden pixel odpovídá 1cm.

Poděkování: Tento materiál vznikl v rámci projektu ESF (CZ.1.07/2.2.00/07.0247)
Reflexe požadavků průmyslu na výuku v oblasti automatického řízení a měření.
Formát zpracování originálu: titulní list barevně, další listy včetně příloh barevně.